Definitions, Acronyms & References

Version 1.1

Edit History

Version	Date	Comments
1.1	26 Jan 2022	Initial Version 1.1 release

Table of Contents

1. Normative references

Information technology -- Dynamic adaptive streaming over HTTP (DASH) -- Part 1: Media presentation description and segment formats	ISO MPEG	ISO/IEC 23009-1:2012	https://www.iso.org/standard/57623.html	Packaging of Video, Audio
Information technology – Dynamic adaptive streaming over HTTP (DASH) – Part 4: Segment encryption and authentication	ISO MPEG	ISO/IEC 23009-4:2018	https://www.iso.org/standard/73603.html	Encryption of MPEG DASH
ISO Base Media Format				
H.264 or MPEG-4 Part 10, Advanced Video Coding (MPEG-4 AVC)	ISO MPEG	ISO/IEC 14496–10, ITU-T H.264	http://www.itu.int/rec/T-REC-H.264	Video Compression
DASH-IF Implementation Guidelines: Content Protection Information Exchange Format (CPIX)	DASH Industry Forum	CPIX	https://dashif.org/docs/DASH-IF-CPIX-v1.0.pdf	Interchange format for exchanging content encryption keys between systems.
Joint Photographic Experts Group	ISO	ISO/IEC 10918, ITU-T T.81, ITU-T T.83, ITU-T T.84, ITU-T T.86	https://jpeg.org/jpeg/	Compression of Still Images
UUID Information technology – Open Systems Interconnection – Procedures for the operation of OSI Registration Authorities: Generation and registration of Universally Unique Identifiers (UUIDs) and their use as ASN.1 object identifier components	ISO	ISO/IEC 9834-8:2005, ITU-T Recommendation X.667	https://www.itu.int/ITU-T/studygroups/com17/oid/X.667-E.pdf https://www.iso.org/standard/36775.html	UUID Format

The OAuth 2.0 Authorization Framework	IETF	RFC 6749	https://tools.ietf.org/html/rfc6749	OAuth Protocol used for token processing
The JavaScript Object Notation (JSON) Data Interchange Format	IETF	RFC 8259	https://tools.ietf.org/html/rfc8259	Definition of JSON
JSON Schema: A Media Type for Describing JSON Documents	IETF	RFC 7159	https://json-schema.org/specification.html	Definition of JSON Schema
The Secure Sockets Layer (SSL) Protocol Version 3.0	IETF	RFC 6101	https://tools.ietf.org/html/rfc6101	HTTPS
JSON Web Encryption	IETF	RFC 7516	https://tools.ietf.org/html/rfc7516	
JSON Web Algorithms	IETF	RFC 7518	https://tools.ietf.org/html/rfc7518	
Key words for use in RFCs to Indicate Requirement Levels	IETF	RFC 2119	https://www.ietf.org/rfc/rfc2119.txt	
Part1 Revision 4 "Recommendation for Key management"	NIST	NIST SP800-57	https://csrc.nist.gov/publications/detail/sp/800-57-part-1/revised/archive/2007-03-01	
Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)	IETF	RFC 6979	https://tools.ietf.org/html/rfc6979	
OAuth 2.0 for Native Apps	IETF	RFC 8252	https://tools.ietf.org/html/rfc8252	

Proof Key for Code Exchange by OAuth Public Clients	IETF	RFC 7636	https://tools.ietf.org/html/rfc7636

2. Terms and Definitions

2.1. Definitions

Term	Definition
Access Token	An Access Token is used by an App or Data Service to access a Device or Data Service. The Access Token conforms to the JSON Web Token format.
Account Management App	An Application provided by the NICE Account Service provider that enables the User to manage their Account. It also manage the linking of Apps and Devices to their Account.
AppInstance	Specific instance of a NICE App.
AppInstanceID	Unique Identifier for the instance of App. This shall conform to the Unique Identifier Specification.
NICE App	A NICE App interacts with a user and utilizes data from the NICE Data Pipeline to provide information and feedback to the end user. The NICE App initiates the configuration of the Data Pipeline by requesting a SceneMode to be implemented based on the capabilities of the Nodes available within the Data Pipeline.
AppID	Unique identifier for an App that may be connected to the NICE system. This AppID refers to the version and name of App and not to the specific instance of an App.

App Instance	An App Instance is a specific instance of the execution of an App that has been developed by an App developer. For example the same App may be downloaded onto two different platforms. The App on each platform shall be referred to as an App Instance.
App Developer	Party that developed an App and operates servers used to distribute data associated with the App.
Black Box	Key server that is used to insert keys and IDs into Devices or chips used in Devices during the production process.
Business Support System	Customers system that is used to manage customer information, processes for onboarding customers and other processes that are proprietary to the customer's operation.
Scene Encryption Key	The Scene Encryption Key shall be used to encrypt or to decrypt SceneData and SceneMarks. They are defined as SceneEncryptionKey in Privacy Object.
Cloud Services	A cloud based Service that interacts with the Device. A Cloud Service may control Devices, consume SceneMarks and SceneData generated by the Device and further process this data to either augment existing SceneMarks and SceneData or generate new SceneMarks and SceneData. A Cloud Service may aggregate and process SceneMarks and SceneData from multiple sources.
Control Session	A connection between an App and a Controller, which enables the App to configure the Nodes which are managed by the Controller.
Controller	A logical function within a Device or Data Service that manages the configuration of one or more Nodes. A Node shall be exclusively managed by a single Controller.
Data Pipeline	The DataPipeline is a workflow for organizing and indexing video in realtime. The DataPipeline comprises of multiple streams from multiple cameras. It is a real time messaging system that provides SceneMarks and SceneData that are relevant for the Application that has configured the DataPipeline. The DataPipeline enables feedback to change parts of the pipeline in response to analysis that has occurred within the DataPipeline – this enables optimized data capture and processed.

Device Certificate	A Certificate that certifies the linkage between a Public Key and a DeviceID. This validates the DeviceID and enables secure communication with the Device. This Certificate is made available once the Device has been registered with a Device Seller.
DeviceID	Identifier for the Device that conforms to the Universally Unique Identifier as defined by ISO/IEC 9834-8:2005. The format of this Identifier is described NICE Identifier Structure specification.
Device Password	Password that is used by the owner of the Device log into the NICE LA account for the Device and enable the Device to be associated with their Account.
End User	The person or entity that owns or manages the Device and has a User Account.
Entity	This is either a Device, Data Service or App within the NICE eco system.
Input Port	A Node uses an Input to receive Data to the Outputs of other Nodes. An Input includes the protocol used for the communication of data, the Universal Resource Indicator for the source of Data from the Node, the encoding of the Data from the Node and the encryption of the Data received by the Node.
SceneEncryptionKeyID	A unique Identifier for a Scene Encryption Key. This Identifier has a value that is unique throughout the NICE System.
Manufacturer	Entity that is responsible for manufacturing a Device.
NICE Account Service	The NICE Account Service manages the User's Account information, the NICE Devices linked to the User's Account and the NICE Apps linked to the User's Account. The NICE Account Service manages the access of NICE Apps to NICE Devices, NICE Data Services to NICE Devices and NICE Apps to NICE Data Services. The NICE Account Service is also responsible for managing the Privacy Management of data associated with the End Users Account including data generated by NICE Devices.

NICE Account ServiceID	Identifier that is allocated to a NICE Account Service by the NICE License Authority. This conforms to the Universally Unique Identifier as defined by ISO/IEC 9834-8:2005.
NICE Data	NICE Data is data that is embodied in the data formats defined in the NICE specification. These include SceneMarks, SceneData and Capabilities.
NICE Data Service	A NICE Data Service is a Service provided to a NICE App which is based on processing the output data stream from the NICE Data Pipeline. A NICE Data Service may enhance this output data or may analyse the data further to create new SceneMarks and SceneData.
NICE Device	A Device is compliant to the NICE Device Specification. It contains at least one Node and implements the Management Interface to the NICE License Authority and NICE Account Services. Each Device has a unique Identifier and can be individually managed by the NICE License Authority and the NICE Account Service. Examples of Physical Devices are Camera, Sensor Devices or Bridges between existing cameras and the NICE eco system. NICE Devices may also be implemented as "Virtual" Devices where the device is implemented as a cloud application.
NICE License Authority	Entity that provides the root of trust for all other Entities in the NICE eco system. The NICE LA provides keys and ids for Devices, NICE Account Services, Apps and Data Services.
NICE Trusted Time Service	The NICE Trusted Time Service is a service provided by either the NICE License Authority or NICE Account Service that is a source of trusted time.
NICE System	The NICE System is the entire ecosystem that covers the NICE Devices, NICE License Authority and NICE Account Service, NICE Data Services, NICE Media Services and NICE Apps.
Node	A Node is the most basic component of the NICE Data Pipeline. A Node has Inputs, Outputs, Transducers (sensors or actuators) and Process's. A Node may capture an image, process its contents and generate a SceneMark and SceneData, it may input audio data and play it back through its speaker (Transducer) or simply take SceneMarks and SceneData as inputs and further process them.

Node Number	The value assigned to a Node within the scope of the Device. The NodeID is the combination of the DeviceID and the Node Number as defined in the NICE Identifier Specification.
Output Port	A Node uses an Output to send Data to the Inputs of other Nodes. An Output includes the protocol used for the communication of data, the Universal Resource Indicator for the destination of Data from the Node, the encoding of the Data from the Node and the encryption of the Data sent from the Node.
Port	A Port can be either an input or an output to a Node. A port can carry video, audio or still and other SceneData into or out of the Node.
Port Number	A value assigned to a Port within the scope of a Node. The PortID is the combination of NodeID and the Port Number as defined in the NICE identifier Specification.
Privacy Agent	The Privacy Agent is a process that is executed in a secure execution environment. It contains the keys that are required to decrypt or encrypt Data and processes the rules that are defined in the Privacy Object.
Privacy Management Service	The system that is used to manage the privacy of Data in the NICE eco system. This includes the Privacy Server that provides Privacy Objects that enable access to the data, the Privacy Agent within the Device and the Privacy Agent within the Data Services and Apps.
Privacy Object	A data object that delivers Scene Encryption Keys that are used to either encrypt or decrypt encrypted SceneData and SceneMarks. The Privacy Object may also define how data that is decrypted shall be handled including whether the data may be analyzed, redistributed. The Privacy Object may also determine the conditions under the which the data may be decrypted.
Privacy Server	The Privacy Server is a cloud service that is part of the NICE Account Service that manages the access to Data through the distribution of Privacy Objects to the Devices, Data Services and Apps to enable them to encrypt, decrypt and process Data.

Private Key	A key that is part of a asymmetric cryptographic key pair. This key is kept secret within the Entity to which the key has been assigned.
Public Key	A key that is part of a asymmetric cryptographic key pair. This key is distributed openly in a X.509 certificate.
SceneData	SceneData is captured or provided by a group of one or more sensor devices and/or sensor modules, which includes different types of sensor data related to the Scene and also further processed or analyzed data. SceneData can be thought of as a sample or snapshot of a Scene. SceneData can also include different types of meta data from various sources. Examples include timestamps, geolocation data, ID for the sensor device, IDs and data from other sensor devices in the vicinity.
SceneMark	A SceneMark is a standardised data structure that describes the Scene that is being captured by the Nodes within the Device. The SceneMark comprises data such as a thumbnail, the settings of the Node, data describing what has been captured in an image frame, a timestamp, references to other SceneMarks related to the Scene. A SceneMark marks the Scene of interest or possibly a point of interest within the Scene.
SceneMarkManifest	The SceneMarkManifest is a data object containing references and URI's to SceneMarks. The SceneMarkManifest is organized by time and may be curated to a specific Apps.
SceneMode	The SceneMode is the configuration of a Node that optimizes the Node to generate data that is tuned to the requirements of the NICE App or NICE Data Service configuring the SceneMode. The SceneMode defines the inputs and outputs, the configuration of the sensor and the processing of data within the Node.
Trusted Time Clock	A clock that executes within an App, Data Service or Device. The clock is resistant to being reset or having its operation modified by any external agent or rogue application within the Device. The clock is initialized by the Trusted Time Protocol.
Trusted Time Stamp	Time stamp that has been signed by either the NICE License Authority or NICE Account Service.

Trusted Time Protocol	This is a protocol defined in the NICE specification that enables a Trusted Time Clock to be synchronized with the master clock within the NICE LA or NICE AS.
Speaker	Transducer that takes as an input a sequence of data and converts them to audible signals.
Transducer	Function of Node that can convert a digital value to a physical output. For example a speaker is a transducer takes a digital input and outputs an audio signal.
Trigger	A field in the SceneMode that defines when a SceneMark shall be generated. For example a trigger condition can be that a face has been detected. When the Node detects a face it shall generate a SceneMark and SceneData.
Trusted Execution Environment	A Trusted Execution Environment is an environment where a sensitive software application can execute and process sensitive data. It shall also have a secure access to encryption and decryption hardware within the Device.
Trusted Time	Time that has been synchronized with the centralized time server using the trusted time protocol defined in the Privacy and Security Specification.
User Account	The NICE User Account is an Account that is used by the owner of one or more NICE Devices can manage their devices, NICE Apps which may access their devices and data. A NICE User Account may belong to a single user, typically a consumer or group of users, typically an enterprise.
Whitelist	A list of DeviceIDs or URIs which the Device is allowed to communicate with.

2.2. Abbreviation

TLS	Transport Level Security
DTLS	Datagram Transport Level Security

NID	Node Identifier
NICELA	NICE License Authority
NICEAS	NICE Account Service
NICEDS	NICE Data Service

2.3. JSON Definitions

```
{
    "$schema": "http://json-schema.org/draft-06/schema#",
    "title": "EndPoint Definition",
    "required": [],
    "definitions": {
        "EndPoint": {
            "type": "object",
            "properties": {
                "AppEndPoint": {
                    "$ref": "#/definitions/ApplicationEndPointSpecifier"
                },
                "NetEndPoint": {
                    "$ref": "#/definitions/NetworkEndPointSpecifier"
                }
            },
            "required": [
                "AppEndPoint",
                "NetEndPoint"
            ]
        },
        "SceneModeConfig": {
            "type": "array",
            "description": "This defines the depth of analysis performed and whether a
resut of an output can be used to drive a subsequent capture of frames.",
            "uniqueItems": true,
            "items": {
                "type": "object",
                "properties": {
                    "AnalysisStage": {
                        "type": "string",
                        "enum": [
                            "Motion",
                            "Detect",
                            "Recognize",
                            "Characterize"
                        ]
                    },
                    "CustomAnalysisStage": {
                        "type": "string",
                        "description": "This defines analysis stages that are
proprietary."
                    },
                    "AnalysisThreshold": {
                        "type": "number",
                        "description": "The output of the analysis should be greater
than this value to trigger the Capture Sequence."
                    },
                    "Scheduling": {
                        "type": "array",
                        "items": {
                            "type": "object",
                            "properties": {
                                "SchedulingType": {
                                    "type": "string",
                                    "enum": [
                                        "Default",
                                        "ScheduledOnce",
                                        "ScheduledHourly",
                                        "ScheduledDaily",
                                        "ScheduledWeekDays",
                                        "ScheduledWeekEndDays",
                                        "ScheduledWeekly",
                                        "ScheduledMonthly",
                                        "ScheduledAnnually"
                                    ]
                                },
                                "StartTime": {
                                    "type": "string"
                                },
                                "EndTime": {
                                    "type": "string"
                                }
                            }
                        }
                    }
                },
                "required": []
            }
        },
        "ApplicationEndPointSpecifier": {
            "type": "object",
            "properties": {
                "APIVersion": {
                    "type": "string",
                    "enum": [
                        "1.0"
                    ]
                },
                "EndPointID": {
                    "type": "string",
                    "description": "The NICE Identifier for the Application that is
ultimatley the end point for messages."
                },
                "X.509Certificate": {
                    "$ref": "#/definitions/x5c"
                },
                "AccessToken": {
                    "type": "string",
                    "description": "This token is used by the receiving NICE entity.
It shall always comply ot the JWT (RFC 7519) format"
                }
            },
            "required": [
                "APIVersion",
                "EndPointID"
            ]
        },
        "NetworkEndPointSpecifier": {
            "type": "object",
            "properties": {
                "APIVersion": {
                    "type": "string",
                    "enum": [
                        "1.0"
                    ]
                },
                "EndPointID": {
                    "type": "string"
                },
                "NodeID": {
                    "type": "string"
                },
                "PortID": {
                    "type": "string"
                },
                "Scheme": {
                    "type": "array",
                    "uniqueItems": true,
                    "items": {
                        "anyOf": [
                            {
                                "$ref": "#/definitions/WebAPIScheme"
                            }
                        ]
                    }
                }
            },
            "required": [
                "APIVersion",
                "EndPointID",
                "Scheme"
            ]
        },
        "WebAPIScheme": {
            "type": "object",
            "title": "Network end point specifier for WebAPI",
            "properties": {
                "Protocol": {
                    "type": "string",
                    "enum": [
                        "WebAPI"
                    ]
                },
                "Authority": {
                    "type": "string"
                },
                "AccessToken": {
                    "type": "string"
                },
                "Role": {
                    "type": "string",
                    "description": "If set to Client, the port shall initiate GET or
SET data requests. If Server then the port shall act as a server. ",
                    "enum": [
                        "Client",
                        "Server"
                    ]
                }
            },
            "required": [
                "Protocol",
                "Authority"
            ]
        },
        "Encryption": {
            "type": "object",
            "properties": {
                "EncryptionOn": {
                    "type": "boolean"
                },
                "SceneEncryptionKeyID": {
                    "type": "string",
                    "description": "Unique Key Identifier that enables the key used to
encrypt the data. If EncryptionOn is False this value will be ignored."
                },
                "PrivacyServerEndPoint": {
                    "type": "object",
                    "properties": {
                        "AppEndPoint": {
                            "$ref": "#/definitions/ApplicationEndPointSpecifier"
                        },
                        "NetEndPoint": {
                            "$ref": "#/definitions/NetworkEndPointSpecifier"
                        }
                    },
                    "required": [
                        "NetEndPoint"
                    ]
                }
            },
            "required": [
                "EncryptionOn"
            ]
        },
        "ThumbnailInfo": {
            "type": "object",
            "additionalProperties": {
                "not": {}
            },
            "properties": {
                "ImageURI": {
                    "type": "string"
                },
                "Encryption": {
                    "$ref": "#/definitions/Encryption"
                }
            },
            "required": [
                "ImageURI",
                "Encryption"
            ]
        },
        "SceneData": {
            "type": "object",
            "description": "For a particular SceneMark there may be several SceneData
objects. This array contains one or more SceneData objects.",
            "properties": {
                "TimeStamp": {
                    "type": "string"
                },
                "SourceNodeID": {
                    "type": "string"
                },
                "SourceNodeDescription": {
                    "type": "string"
                },
                "Duration": {
                    "type": "string"
                },
                "DataType": {
                    "type": "string",
                    "description": "Types of data that is in the SceneData object.",
                    "enum": [
                        "RGBStill",
                        "IRStill",
                        "DepthStill",
                        "RGBStereoStill",
                        "ThermalStill",
                        "RGBVideo",
                        "IRVideo",
                        "DepthVideo",
                        "RGBStereoVideo",
                        "ThermalVideo"
                    ]
                },
                "Status": {
                    "type": "string",
                    "enum": [
                        "Available at Provided URI",
                        "Upload in Progress"
                    ]
                },
                "MediaFormat": {
                    "type": "string",
                    "enum": [
                        "JPEG",
                        "H.264",
                        "H.265",
                        "RAW",
                        "JSON"
                    ]
                },
                "Resolution": {
                    "type": "object",
                    "properties": {
                        "Height": {
                            "type": "integer"
                        },
                        "Width": {
                            "type": "integer"
                        }
                    }
                },
                "SceneDataID": {
                    "type": "string",
                    "description": "Unique Identifier for the SceneData referenced by
this data structure."
                },
                "SceneDataURI": {
                    "type": "string",
                    "description": "This is URI to an external data object."
                },
                "EmbeddedSceneData": {
                    "type": "string",
                    "description": "Data may be directly embedded in the SceneMark.
The Data is encoded as Base64."
                },
                "Encryption": {
                    "$ref": "#/definitions/Encryption"
                }
```

```
            },
            "required": [
                "TimeStamp",
                "Encryption",
                "SceneDataID"
            ]
        },
        "VersionControl": {
            "type": "object",
            "properties": {
                "FinalEntityID": {
                    "type": "string",
                    "description": "The Application or DataService that will receive
the SceneMark."
                },
                "ProcessList": {
                    "type": "array",
                    "description": "List of Nodes that have processed SceneMark in the
order of processing. ",
                    "items": {
                        "type": "object",
                        "properties": {
                            "NodeID": {
                                "type": "string",
                                "description": "NodeID that has processed the
SceneMark"
                            },
                            "TimeStamp": {
                                "type": "string",
                                "description": "TimeStamp of when the Node processed
the SceneMark"
                            },
                            "Vector": {
                                "type": "string",
                                "description": "Hash of status code for node"
                            }
                        },
                        "required": [
                            "NodeID"
                        ]
                    }
                }
            }
        },
        "x5c": {
            "type": "array",
            "description": "X.509 certification chain that enables the receiver to
verify the entities (Device, App, DataService, NICE AS etc) against the Root
Certificate for NICE. Each array item is a certificate in the certificate hierarchy,
Base64 encoded as defined in RFC 7517.\n",
            "items": {
                "type": "string"
            }
        },
        "SceneMode": {
            "type": "object",
            "properties": {
                "Version": {
                    "type": "string",
                    "enum": [
                        "1.0"
                    ]
                },
                "SceneModeID": {
                    "type": "string",
                    "description": "SceneModeID unique within the scope of this node."
                },
                "NodeID": {
                    "type": "string"
                },
                "Inputs": {
                    "type": "array",
                    "description": "Defines inputs for the Node from other Nodes
either within the device or external devices. String contains URI for the location of
the input.\n",
                    "uniqueItems": true,
                    "items": {
                        "type": "object",
                        "properties": {
                            "PortID": {
                                "type": "string",
                                "description": "ID of the Port which the input is
made. This reference is used by functions in the Node to reference the Input."
                            },
                            "EndPoint": {
                                "$ref":
"Definitions.json#/definitions/NetworkEndPointSpecifier"
                            }
                        },
                        "required": [
                            "PortID"
                        ]
                    }
                },
                "Outputs": {
                    "type": "array",
                    "description": "Defines outputs including data encoding, protocols
and end points.",
                    "uniqueItems": true,
                    "items": {
                        "anyOf": [
                            {
                                "type": "object",
                                "title": "VideoConfiguration",
                                "description": "This configures the output of the
video output of the Node",
                                "properties": {
                                    "Type": {
                                        "type": "string",
                                        "enum": [
                                            "Video"
                                        ]
                                    },
                                    "PortID": {
                                        "type": "string",
                                        "description": "ID allocated within the Node
for this output. "
                                    },
                                    "FrameRate": {
                                        "type": "number",
                                        "description": "Frames per second"
                                    },
                                    "Resolution": {
                                        "type": "object",
                                        "description": "Resolution in pixels.",
                                        "properties": {
                                            "Height": {
                                                "type": "integer"
                                            },
                                            "Width": {
                                                "type": "integer"
                                            }
                                        },
                                        "required": [
                                            "Height",
                                            "Width"
                                        ]
                                    },
                                    "DestinationEndPointList": {
                                        "type": "array",
                                        "uniqueItems": true,
                                        "items": {
                                            "$ref":
"Definitions.json#/definitions/EndPoint"
                                        }
                                    },
                                    "MediaFormat": {
                                        "type": "string",
                                        "enum": [
                                            "JPEG",
                                            "H.264",
                                            "H.265",
                                            "RAW"
                                        ]
                                    },
                                    "StartTimeRelTrigger": {
                                        "type": "number"
                                    },
                                    "EndTimeRelTrigger": {
                                        "type": "number"
                                    },
                                    "MaxChunkSize": {
                                        "type": "number"
                                    }
                                },
                                "required": [
                                    "PortID",
                                    "Type"
                                ]
                            },
                            {
                                "type": "object",
                                "title": "DataStreamConfig",
                                "description": "This condigures a data stream from
sensors such as thermometer, CO2 etc.",
                                "properties": {
                                    "Type": {
                                        "type": "string",
                                        "enum": [
                                            "Humidity",
                                            "CarbonMonoxide",
                                            "PIR",
                                            "Temperature",
                                            "IRDetection",
                                            "Pressure",
                                            "Proximity",
                                            "LiquidLevel",
                                            "Acceleration",
                                            "Rotation"
                                        ]
                                    },
                                    "PortID": {
                                        "type": "string",
                                        "description": "PortID allocated within the
Node for this output. "
                                    },
                                    "SamplePeriod": {
                                        "type": "number",
                                        "description": "Time between samples. 1 means
1 sample per second. O.5 means 2 samples per second. 10 means one sample every 10
seconds."
                                    },
                                    "MediaFormat": {
                                        "type": "string",
                                        "enum": [
                                            "JSON"
                                        ]
                                    },
                                    "DestinationEndPointList": {
                                        "type": "array",
                                        "uniqueItems": true,
                                        "items": {
                                            "$ref":
"Definitions.json#/definitions/NetworkEndPointSpecifier"
                                        }
                                    },
                                    "StartTimeRelTrigger": {
                                        "type": "number"
                                    },
                                    "EndTimeRelTrigger": {
                                        "type": "number"
                                    },
                                    "MaxChunkSize": {
                                        "type": "number"
                                    }
                                },
                                "required": [
                                    "PortID",
                                    "Type"
                                ]
                            },
                            {
                                "type": "object",
                                "title": "AudioStreamConfiguration",
                                "description": "Configures audio stream output from
Node.",
                                "properties": {
                                    "Type": {
                                        "type": "string",
                                        "enum": [
                                            "Audio"
                                        ]
                                    },
                                    "PortID": {
                                        "type": "string"
                                    },
                                    "SampleRate": {
                                        "type": "number"
                                    },
                                    "SampleDepth": {
                                        "type": "integer"
                                    },
                                    "MediaFormat": {
                                        "type": "string",
                                        "enum": [
                                            "AAC",
                                            "MP3",
                                            "WAV"
                                        ]
                                    },
                                    "DestinationEndPointList": {
                                        "type": "array",
                                        "description": "Defines the locations that the
audio stream may be writen to. This may be using WebRTC or HTTPS.",
                                        "uniqueItems": true,
                                        "items": {
                                            "$ref":
"Definitions.json#/definitions/NetworkEndPointSpecifier"
                                        }
                                    },
                                    "StartTimeRelTrigger": {
                                        "type": "number"
                                    },
                                    "EndTimeRelTrigger": {
                                        "type": "number"
                                    },
                                    "MaxChunkSize": {
                                        "type": "number"
                                    }
                                },
                                "required": [
                                    "PortID",
                                    "Type"
                                ]
                            },
                            {
                                "type": "object",
                                "title": "ImageConfiguration",
                                "description": "Image output configuration",
                                "properties": {
                                    "Type": {
                                        "type": "string",
                                        "enum": [
                                            "Image"
                                        ]
                                    },
                                    "PortID": {
                                        "type": "string"
                                    },
                                    "ImageType": {
                                        "type": "string",
                                        "enum": [
                                            "RGB",
                                            "IR",
                                            "RGBIR",
                                            "Depth",
                                            "Thermal",
                                            "StereoRGB"
                                        ]
                                    },
                                    "Resolution": {
                                        "type": "object",
                                        "description": "Resolution in pixels.",
                                        "properties": {
                                            "Height": {
                                                "type": "integer"
                                            },
                                            "Width": {
                                                "type": "integer"
                                            }
                                        },
                                        "required": [
                                            "Height",
                                            "Width"
                                        ]
                                    },
                                    "MediaFormat": {
                                        "type": "string",
                                        "enum": [
                                            "JPEG",
                                            "RAW"
                                        ]
                                    },
                                    "DestinationEndPointList": {
                                        "type": "array",
                                        "uniqueItems": true,
                                        "items": {
                                            "$ref":
"Definitions.json#/definitions/EndPoint"
                                        }
                                    },
                                    "TimeRelTrigger": {
                                        "type": "number"
                                    }
                                },
                                "required": [
                                    "PortID",
                                    "Type"
                                ]
                            }
                        ]
                    }
                },
                "Transducers": {
                    "type": "array",
                    "uniqueItems": true,
                    "items": {
                        "anyOf": [
                            {
                                "type": "object",
                                "title": "AudioSensorConfig",
                                "description": "Configuration of the audio sensor.",
                                "properties": {
                                    "Type": {
                                        "type": "string",
                                        "enum": [
                                            "Microphone"
                                        ]
                                    },
                                    "TransducerID": {
                                        "type": "string",
                                        "description": "Unique ID for the sensor."
                                    },
                                    "SampleRate": {
                                        "type": "number",
                                        "description": "Number of samples per second."
                                    },
                                    "SampleDepth": {
                                        "type": "integer",
                                        "description": "Number of bits of resolution
for audio samples."
                                    },
                                    "InputToAnalysis": {
                                        "type": "boolean",
                                        "description": "If true then Audio is sent to
the Analysis function in the Node."
                                    },
                                    "PortIDList": {
                                        "type": "array",
                                        "description": "The audio should be sent to
the listed PortIDs",
                                        "uniqueItems": true,
                                        "items": {
                                            "type": "string"
                                        }
                                    }
                                },
                                "required": [
                                    "InputToAnalysis",
                                    "TransducerID",
                                    "Type"
                                ]
                            },
                            {
                                "type": "object",
                                "description": "The volume of the speaker can be
controlled. ",
                                "properties": {
                                    "Type": {
                                        "type": "string",
                                        "enum": [
                                            "Speaker"
                                        ]
                                    },
                                    "TransducerID": {
                                        "type": "string",
                                        "description": "Unque ID of the speaker."
                                    },
                                    "OutputLevel": {
                                        "type": "number",
                                        "description": "Output level setting in dB."
                                    },
                                    "PortID": {
                                        "type": "string",
                                        "description": "Port with incomming audio
stream."
                                    }
                                },
                                "required": [
                                    "PortID",
                                    "TransducerID",
                                    "Type"
                                ]
                            },
                            {
                                "type": "object",
                                "description": "ID of the Capture Sequence that should
be used for the configuration of the Image Sensor. If no CaptureSequence is defined
the Node may use a default setting associated by the implementation of the Node. ",
                                "properties": {
                                    "Type": {
                                        "type": "string",
                                        "enum": [
                                            "Image Sensor"
                                        ]
                                    },
                                    "TransducerID": {
                                        "type": "string"
                                    },
                                    "DefaultCaptureSequence": {
                                        "type": "string"
                                    },
                                    "InputToAnalysis": {
                                        "type": "boolean",
                                        "description": "If true then input to Analysis
function."
                                    },
                                    "PortIDList": {
                                        "type": "array",
                                        "description": "",
                                        "uniqueItems": true,
                                        "items": {
                                            "type": "string"
                                        }
                                    }
                                },
                                "required": [
                                    "InputToAnalysis",
                                    "TransducerID",
                                    "Type"
                                ]
                            }
                        ]
                    }
                },
                "Mode": {
                    "type": "object",
                    "description": "Defines either NICE defined SceneMode or
DeviceDefinedAnalysis - which is a proprietary Computer Vision or AI in the device.",
                    "properties": {
                        "SceneMode": {
                            "type": "string",
                            "enum": [
                                "Motion",
                                "Face",
                                "Human",
                                "Vehicle",
                                "Label",
                                "Animal",
                                "TexLogoQRCode",
                                "Custom"
                            ]
                        },
                        "SceneModeConfig": {
                            "type": "array",
                            "description": "This defines the depth of analysis
performed and whether a resut of an output can be used to drive a subsequent capture
of frames.",
                            "uniqueItems": true,
                            "items": {
                                "type": "object",
                                "properties": {
                                    "AnalysisStage": {
                                        "type": "string",
                                        "enum": [
                                            "Motion",
                                            "Detect",
                                            "Recognize",
                                            "Characterize"
                                        ]
                                    },
                                    "AnalysisDrivenRegionOfInterest": {
                                        "type": "boolean",
                                        "description": "If TRUE then the regions of
interest generated from this analysis stage should be fed back to another Node."
                                    },
                                    "AnalysisThreshold": {
                                        "type": "number",
                                        "description": "The output of the analysis
should be greater than this value to trigger the Capture Sequence."
                                    },
                                    "CaptureSequenceID": {
                                        "type": "string",
                                        "description": "Capture Sequence ID to be
implemented if the analsyis has a positive outcome"
                                    },
                                    "FeedbackEndPoint": {
                                        "$ref":
"Definitions.json#/definitions/NetworkEndPointSpecifier"
                                    }
                                },
                                "required": []
                            }
                        },
                        "CustomAnalysis": {
                            "type": "object",
                            "description": "Algorithm implemented in device that
performs a proprietary AI function.",
                            "properties": {
                                "CustomAnalysisID": {
                                    "type": "string",
                                    "description": "Each algorithm and set of weights
has a unique ID that is defined by NICE. This value shall be carried in this record."
                                },
                                "AnalysisDescription": {
                                    "type": "string",
                                    "description": "Description of algorithm."
                                },
                                "AnalysisDrivenRegionofInterest": {
                                    "type": "boolean",
                                    "description": "If TRUE then the regions of
interest generated from this analysis stage should be fed back to another Node."
                                },
                                "AnalysisThreshold": {
                                    "type": "number",
                                    "description": "The output of the analysis should
be greater than this value to trigger the Capture Sequence."
                                },
                                "CaptureSequenceID": {
                                    "type": "string",
                                    "description": "Capture Sequence ID to be
implemented if the analsyis has a positive outcome"
                                },
                                "FeedbackEndPoint": {
                                    "$ref":
"Definitions.json#/definitions/NetworkEndPointSpecifier"
                                },
                                "Encryption": {
                                    "$ref": "Definitions.json#/definitions/Encryption"
                                }
                            }
                        },
                        "LabelRefDataList": {
                            "type": "array",
                            "description": "For a specific label the following are
reference data such as images for the particular label. The Node shall process these
images to create the appropriate reference vector and store which RefDataIDs have been
used to create the vector. If new RefDataIDs are detected in the SceneMode object the
vector shall be regenerated with the listed RefData.",
                            "uniqueItems": true,
                            "items": {
                                "type": "object",
                                "properties": {
                                    "LabelName": {
                                        "type": "string",
                                        "description": "Label name for example for
facial recognition this would be the name or id of an individual."
                                    },
                                    "RefDataList": {
                                        "type": "array",
                                        "uniqueItems": true,
                                        "items": {
                                            "type": "object",
                                            "properties": {
                                                "RefDataID": {
                                                    "type": "string"
                                                },
                                                "RefDataEndPoint": {
                                                    "$ref":
"Definitions.json#/definitions/NetworkEndPointSpecifier"
                                                }
                                            },
                                            "required": [
                                                "RefDataID"
                                            ]
                                        }
                                    },
                                    "RefData": {
                                        "type": "array",
                                        "uniqueItems": true,
                                        "items": {
                                            "type": "object",
                                            "properties": {
                                                "RefDataID": {
                                                    "type": "string"
                                                },
                                                "RefData": {
                                                    "type": "string",
                                                    "description": "Reference data
encoded in Base64. For example an image of a persons face."
                                                },
                                                "Encryption": {
                                                    "$ref":
"Definitions.json#/definitions/Encryption"
                                                }
                                            },
                                            "required": [
                                                "RefDataID",
                                                "RefData",
                                                "Encryption"
                                            ]
                                        }
                                    },
                                    "ProcessingStage": {
                                        "type": "string",
                                        "description": "This indicates which analysis
stage should use the reference data.",
                                        "enum": [
                                            "CustomAnalysis",
                                            "Motion",
                                            "Detect",
                                            "Recognize",
                                            "Characterize"
                                        ]
                                    }
                                },
                                "required": [
                                    "ProcessingStage",
                                    "LabelName"
                                ]
                            }
                        },
                        "SceneMarkInputList": {
                            "type": "array",
                            "description": "Locations from where incomming SceneMarks
may be received.",
                            "uniqueItems": true,
                            "items": {
                                "type": "object",
                                "properties": {
                                    "SceneMarkInputEndPoint": {
                                        "$ref":
"Definitions.json#/definitions/NetworkEndPointSpecifier"
                                    }
                                },
                                "required": []
                            }
                        },
                        "SceneMarkOutputList": {
                            "type": "array",
                            "description": "Locations where SceneMarks shoudl be sent
to",
                            "uniqueItems": true,
                            "items": {
                                "type": "object",
                                "properties": {
                                    "SceneMarkOutputEndPoint": {
                                        "$ref":
"Definitions.json#/definitions/NetworkEndPointSpecifier"
                                    }
                                },
                                "required": []
                            }
                        },
                        "AudioAnalysisID": {
                            "type": "string"
                        },
                        "TransducerInput": {
                            "type": "array",
                            "description": "ID of Transducer Within Node to be used as
Input to Analysis",
                            "uniqueItems": true,
                            "items": {
                                "type": "string"
                            }
                        }
                    },
                    "required": [
                        "SceneMode"
                    ]
                }
            },
            "required": [
                "Version",
                "NodeID"
            ]
        },
        "SceneEncryptionKey": {
            "type": "object",
            "properties": {
                "alg": {
                    "type": "string",
                    "description": "Indicates for which algorithm the key is intended
to be used for."
                },
                "k": {
                    "type": "string",
                    "description": "Random 256 bit key value encoded as Base64URL"
                },
                "iv": {
                    "type": "string",
                    "description": "Initialization Vector - random 256 bits encoded in
Base64URL"
                },
                "kid": {
                    "type": "string",
                    "description": "Key Identifier which is unique for every
SceneEncryptionKey."
                }
            },
            "required": [
                "alg",
                "k",
                "kid"
            ]
        }
    }
}
```