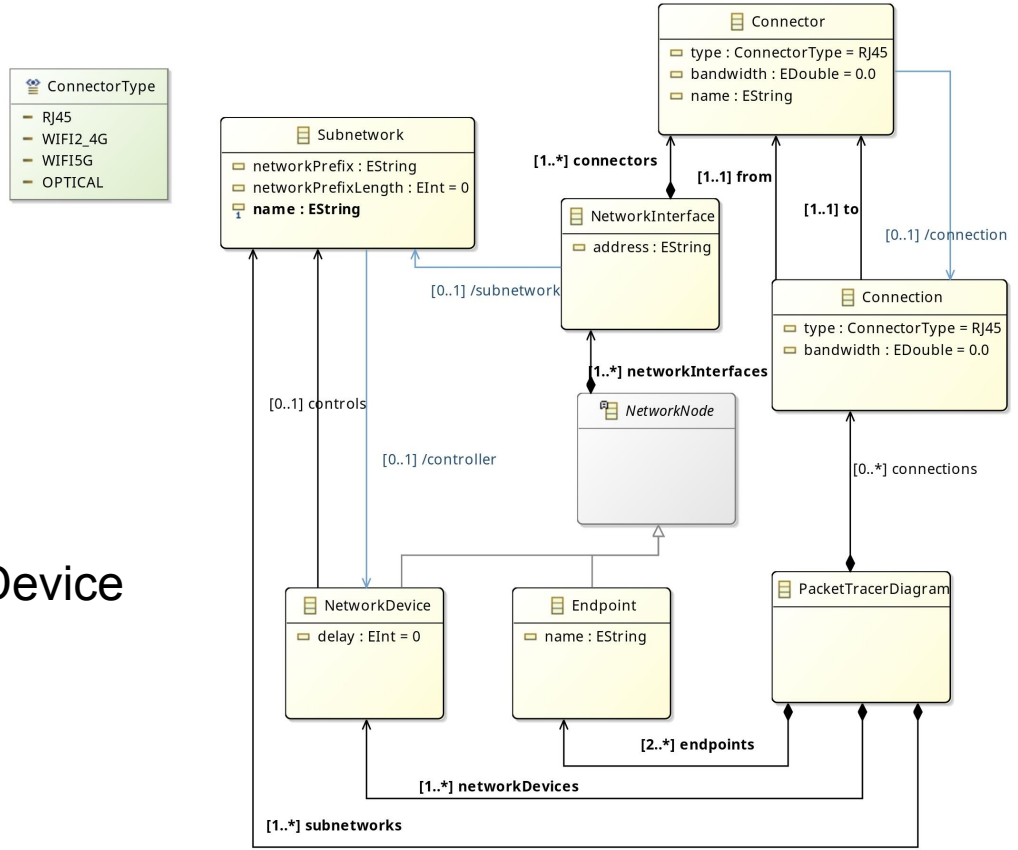
# Plastic Turtle Militia Packet tracer diagram

Antal János Benjamin G9PTHG
Gacsályi Márton LD8CAO

# Metamodel

- NetworkNode
  - NetworkDevice (router)
  - Endpoint (computer)
- Subnetwork
  - Controlled by a NetworkDevice
- NetworkInterface
  - Connector
- Connection

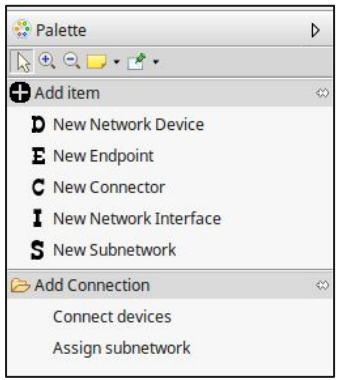


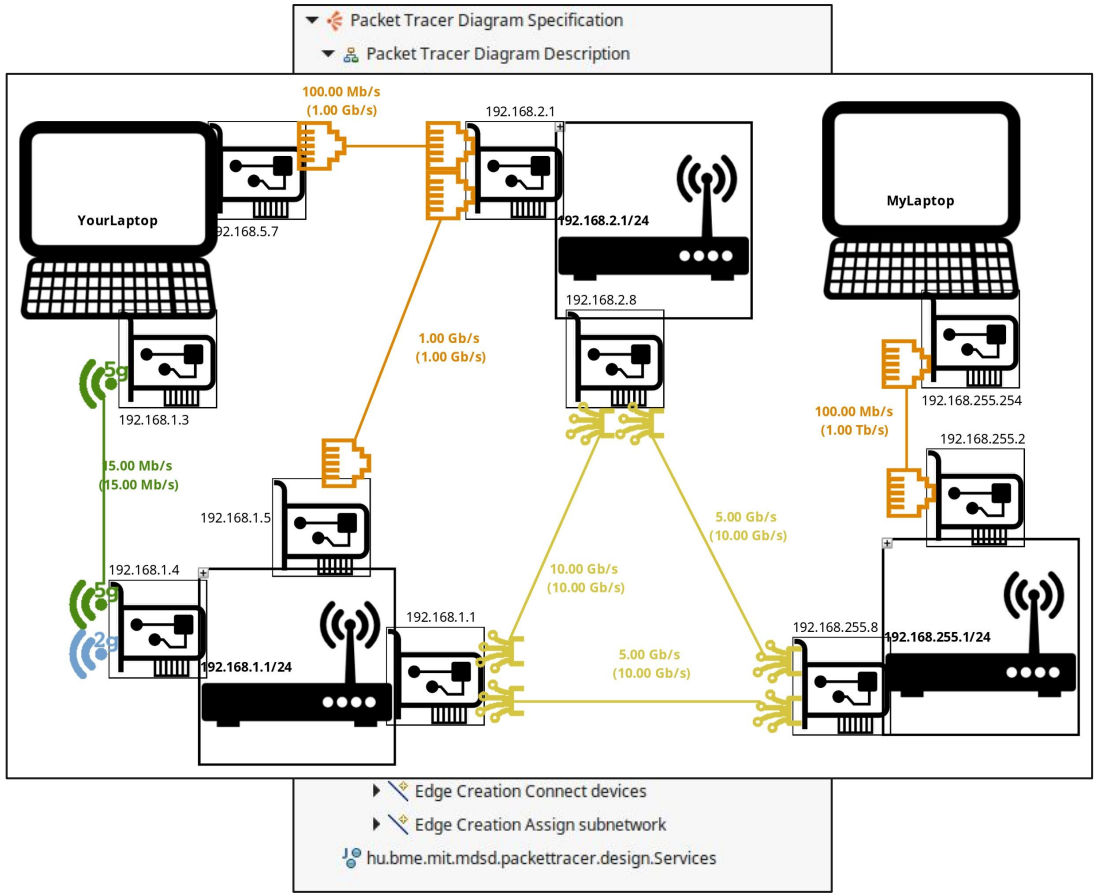
# Xtext editor

- YAML/JSON like
- Mostly derived from the metamodel
- Couldn’t integrate with other tools

```
PacketTracerDiagram {
    networkDevices {
        NetworkDevice {
            delay 5
            controls subnet
            networkInterfaces {
                NetworkInterface {
                    address '192.168.1.1'
                    connectors {
                        Connector con1 { type RJ45 bandwidth 80.0 }
                    }
                }
            }
        }
    } endpoints {
        Endpoint MyLaptop {
            networkInterfaces {
                NetworkInterface {
                    address "192.168.1.2"
                    connectors {
                        Connector con2 { type RJ45 bandwidth 80.0 }
                    }
                }
            }
        }, Endpoint YourLaptop {
            networkInterfaces {
                NetworkInterface {
                    address "192.168.1.3"
                    connectors {
                        Connector con3 { type RJ45 bandwidth 80.0 }
                    }
                }
            }
        }
    } connections {
        Connection { type RJ45 bandwidth 80.0 from con1 to con1 }
    } subnetworks {
        Subnetwork subnet{
            networkPrefix "192.168.1.1"
            networkPrefixLength 24
        }
    }
}
```

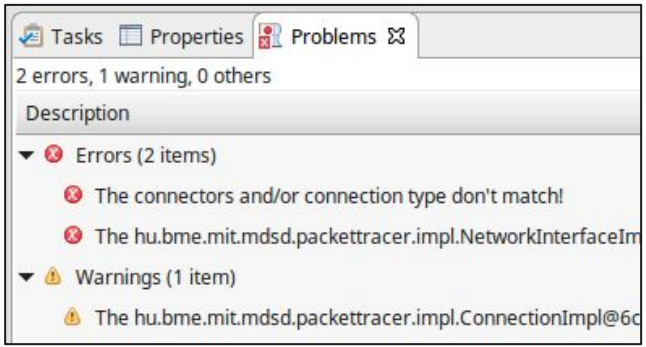
# Sirius editor



# VIATRA Queries

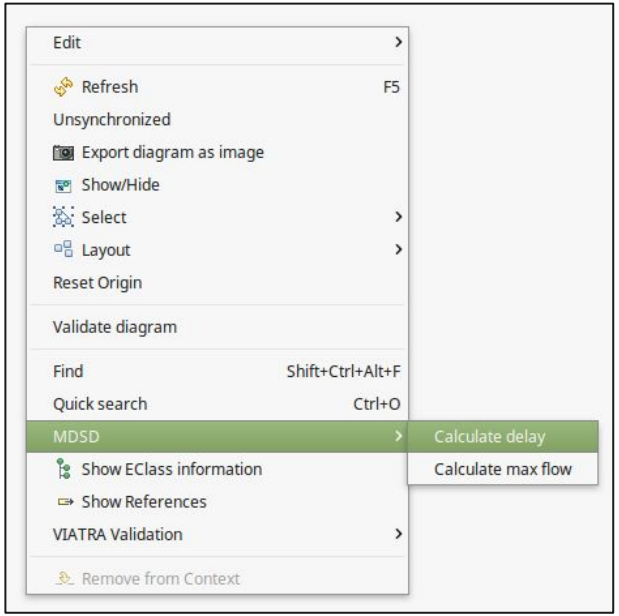
- Constraints (10)
- Warnings (1)
- Derived features (3)
- Make easier to query the model (12)
- A lot of query…

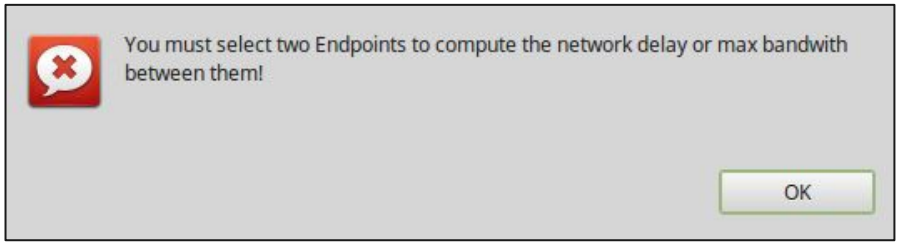


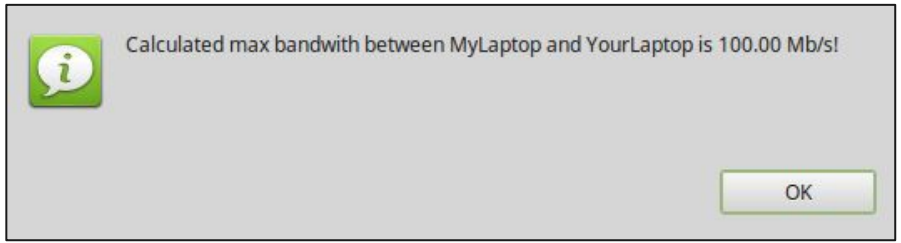

```
@Constraint(key = {
    networkInterface
}, severity = "error", message = "The $networkInterface$($networkInterface.addr
pattern
ipAddressIsOutOfSubnetwork(networkInterface : NetworkInterface, subnetwork : Su
    Subnetwork.networkPrefix(subnetwork, subnetIP);
    Subnetwork.networkPrefixLength(subnetwork, maskLength);
    find subnetwork(networkInterface, subnetwork);
    NetworkInterface.address(networkInterface, ipAddress);
    check(! QueryHelperFunctions.isInSubnet(subnetIP, maskLength, ipAddress));
}

@QueryBasedFeature
pattern subnetwork(interface : NetworkInterface, subnetwork : Subnetwork) {
    Endpoint.networkInterfaces(_, interface);
    NetworkInterface.connectors(interface, connector);
    find connected(connector, otherConnector);
    connector != otherConnector;
    NetworkInterface.connectors(otherInterface, otherConnector);
    NetworkDevice.networkInterfaces(device, otherInterface);
    NetworkDevice.controls(device, subnetwork);
} or {
    NetworkDevice.networkInterfaces(networkDevice, interface);
    NetworkDevice.controls(networkDevice, subnetwork);
}

@QueryBasedFeature
pattern controller(subnetwork : Subnetwork, networkDevice : NetworkDevice) {
    NetworkDevice.controls(networkDevice, subnetwork);
}

@QueryBasedFeature
pattern connection(connector : Connector, connection : Connection) {
    Connection.from(connection, connector);
} or {
    Connection.to(connection, connector);
}
```

# Computations

- Network delay between two endpoints
  - Dijsktra algorithm
- Maximum bandwith between two endpoints
  - Ford–Fulkerson algorithm
- Work in Sirius and Ecore editor too
- Dialog to show error or result



# Lessons learnt

Bad ones:

- VIATRA could have a better format
- Xtext and Sirius integration is not easy (Csaba was right)
- Clean&Restart can help

Good ones:

- Create new DSLs are quite easy
- VIATRA is easy to use
- Sirius isn’t really bad, indeed it’s quite easy too