Application

Notes and Ideas

The network intercetion I am thinking will be an array inside the application itself. The network will need a list of application ids it is associated with I think but probably not more than that.

Requirements

- Should allow a single application to support multiple integrations

API and Data Shape

Create Application

Note: `integrations` and `networks` is optional.

```
POST /api/applications
```

Body

```
{
  "name": "appName",
  "description": "appDescription",
  "payloadCodec": "string",
  "payloadDecoderScript": "string",
  "payloadEncoderScript": "string",
  "validationScript": "string",
  "supportsDownLink": "boolean",
  "running": "boolean",
  "applicationEUI": "string",
  "customerId": "string",
  "integrations": [
    {
      "baseUrl": "baseUrl",
      "reportingProtocolId": "reportingProtocolId"
    }
  ],
  "networkDeployments": [
    "networkId1",
    "networkId2"
  ]
}
```

Returns 201

```
{}
```

Get Application

```
GET /api/applications/:id
```

Returns 200

```
{
  "id": "string",
  "name": "appName",
  "description": "appDescription",
  "payloadCodec": "string",
  "payloadDecoderScript": "string",
  "payloadEncoderScript": "string",
  "validationScript": "string",
  "supportsDownLink": "boolean",
  "running": "boolean",
  "applicationEUI": "string",
  "customerId": "string",
  "integrations": [
    {
      "baseUrl": "baseUrl",
      "reportingProtocolId": "reportingProtocolId"
    }
  ],
  "networkDeployments": [
    {
        "networkId": "string",
        "remoteApplicationId": "string",
        "serviceProfileId": "string",
        "organizationId": "string",
        "securityData": {
            "accessToken": "string",
            "refreshToken": "string"
        }
    }
    ]
}
```

Get Many Applications

```
GET /api/applications?query=q
```

Returns 200

```
{
  "returned": "number",
  "available": "number",
  "results": [
    {
      "id": "string",
      "name": "appName",
      "description": "appDescription",
      "running": "boolean",
      "customerId": "string",
      "networkDeployments": ["networkId1", "networkId2"]
    }]
}
```

Update Application

```
PUT /api/applications/:id
```

Body

```
{
  "name": "appName",
  "description": "appDescription",
  "payloadCodec": "string",
  "payloadDecoderScript": "string",
  "payloadEncoderScript": "string",
  "validationScript": "string",
  "supportsDownLink": "boolean",
  "running": "boolean",
  "applicationEUI": "string",
  "customerId": "string",
  "integrations": [
    {
      "baseUrl": "baseUrl",
      "reportingProtocolId": "reportingProtocolId"
    }
  ],
  "networkDeployments": [
    "networkId1",
    "networkId2"
  ]
}
```

Return 200

```
{
  "id": "string",
  "name": "appName",
  "description": "appDescription",
  "payloadCodec": "string",
  "payloadDecoderScript": "string",
  "payloadEncoderScript": "string",
  "validationScript": "string",
  "supportsDownLink": "boolean",
  "running": "boolean",
  "applicationEUI": "string",
  "customerId": "string",
  "integrations": [
    {
      "baseUrl": "baseUrl",
      "reportingProtocolId": "reportingProtocolId"
    }
  ],
  "networkDeployments": [
    {
        "networkId": "string",
        "remoteApplicationId": "string",
        "serviceProfileId": "string",
        "organizationId": "string",
        "securityData": {
            "accessToken": "string",
            "refreshToken": "string"
        }
    }
    ]
}
```

Delete Application

```
DELETE /api/applications/:id
```

Body

```
{}
```

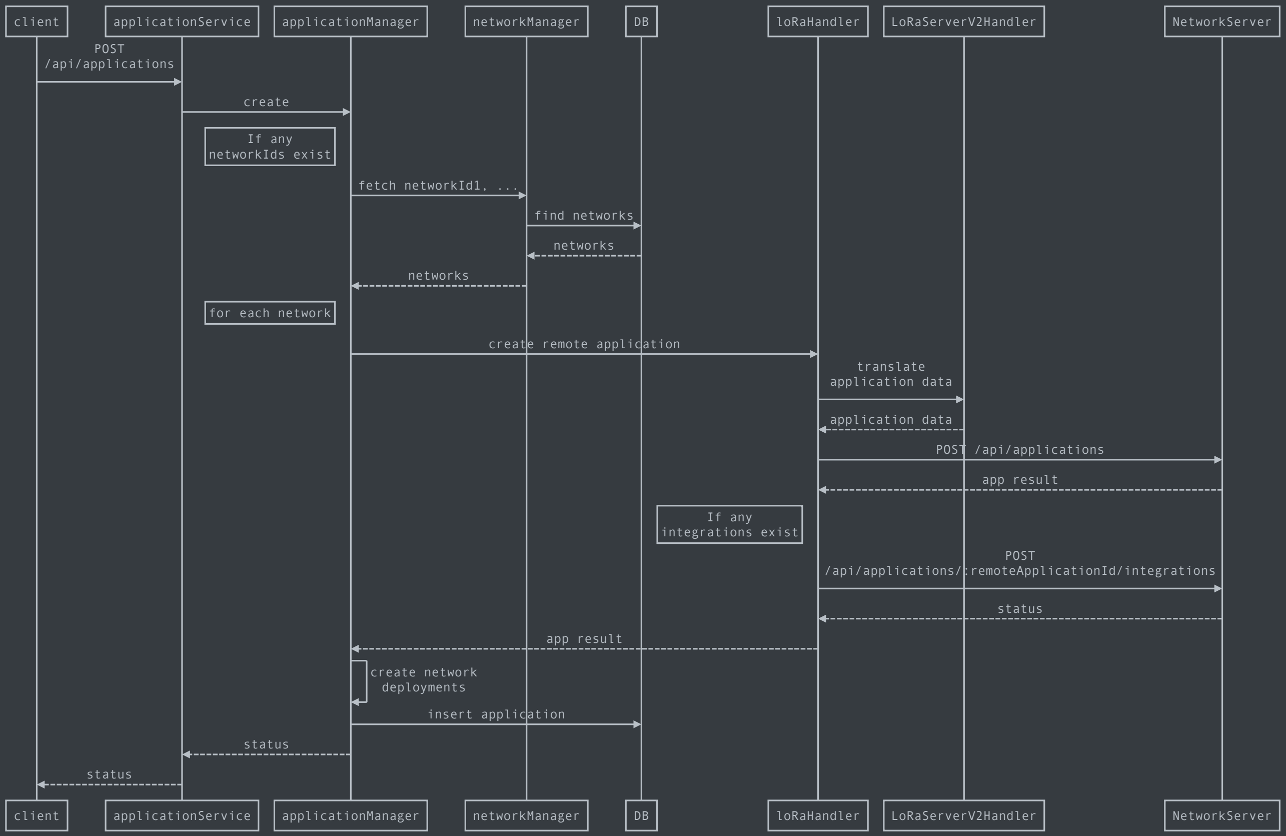
Return 204

```
{}
```

Proposed Flow

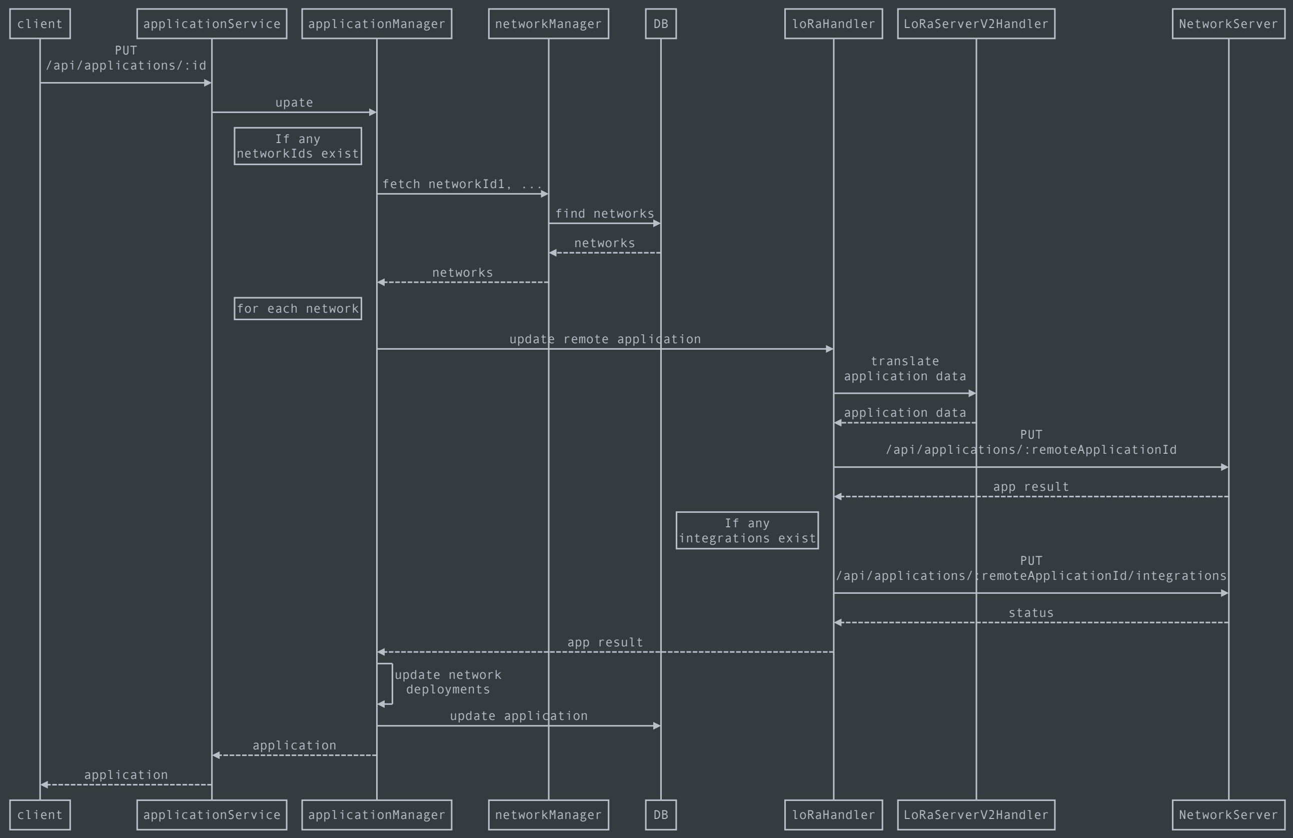
New Application

Note: Used LoRaServerV2 as an example, could be any protocol

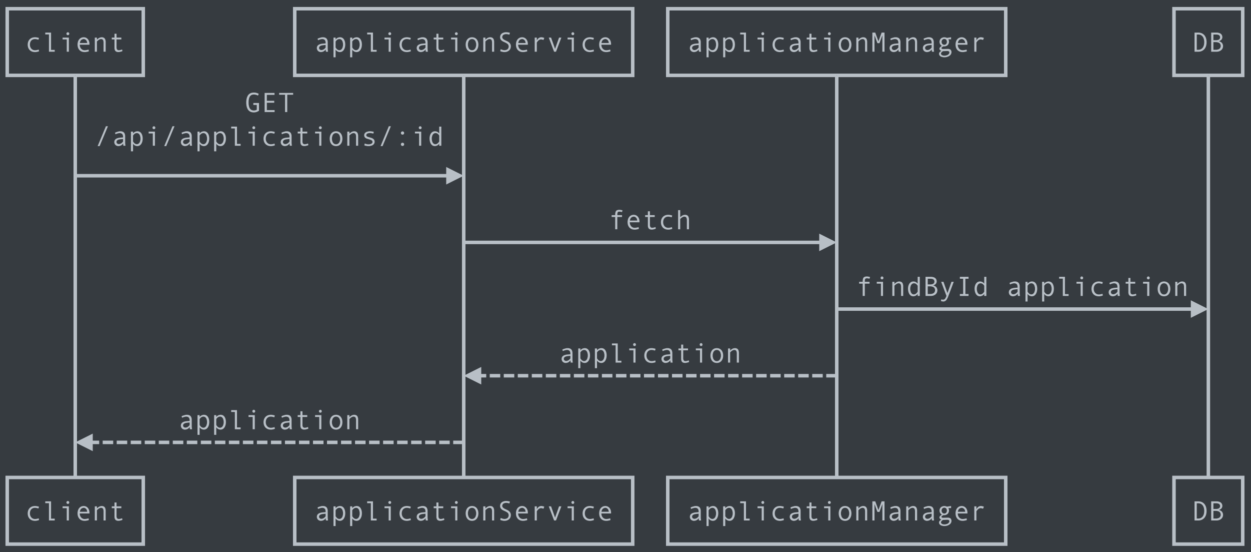


Update Application

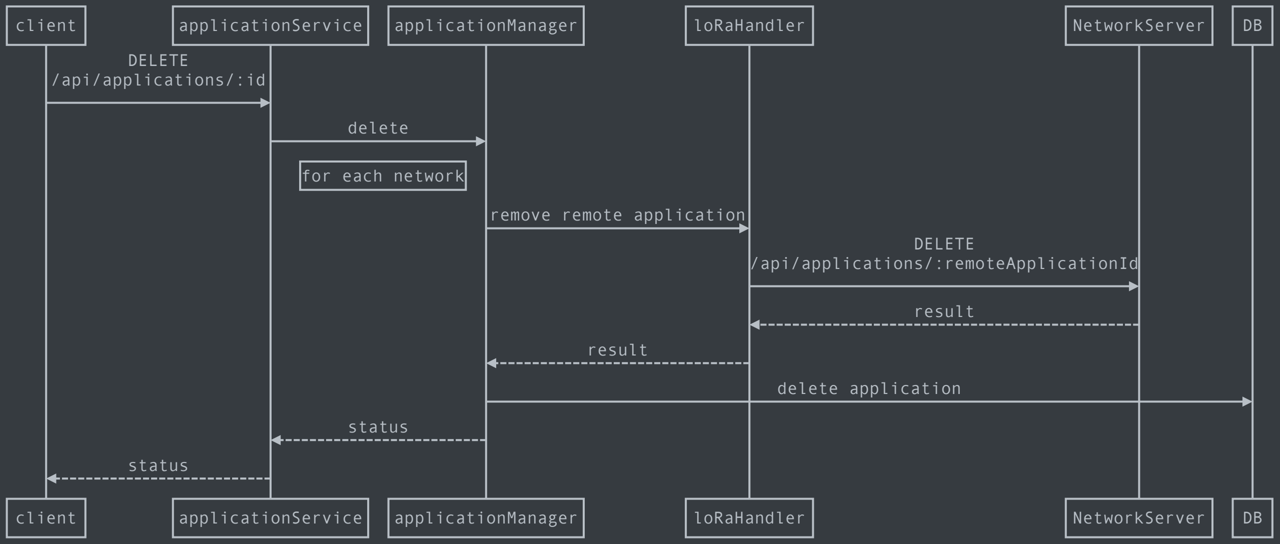
Note: Used LoRaServerV2 as an example, could be any protocol



Get Application

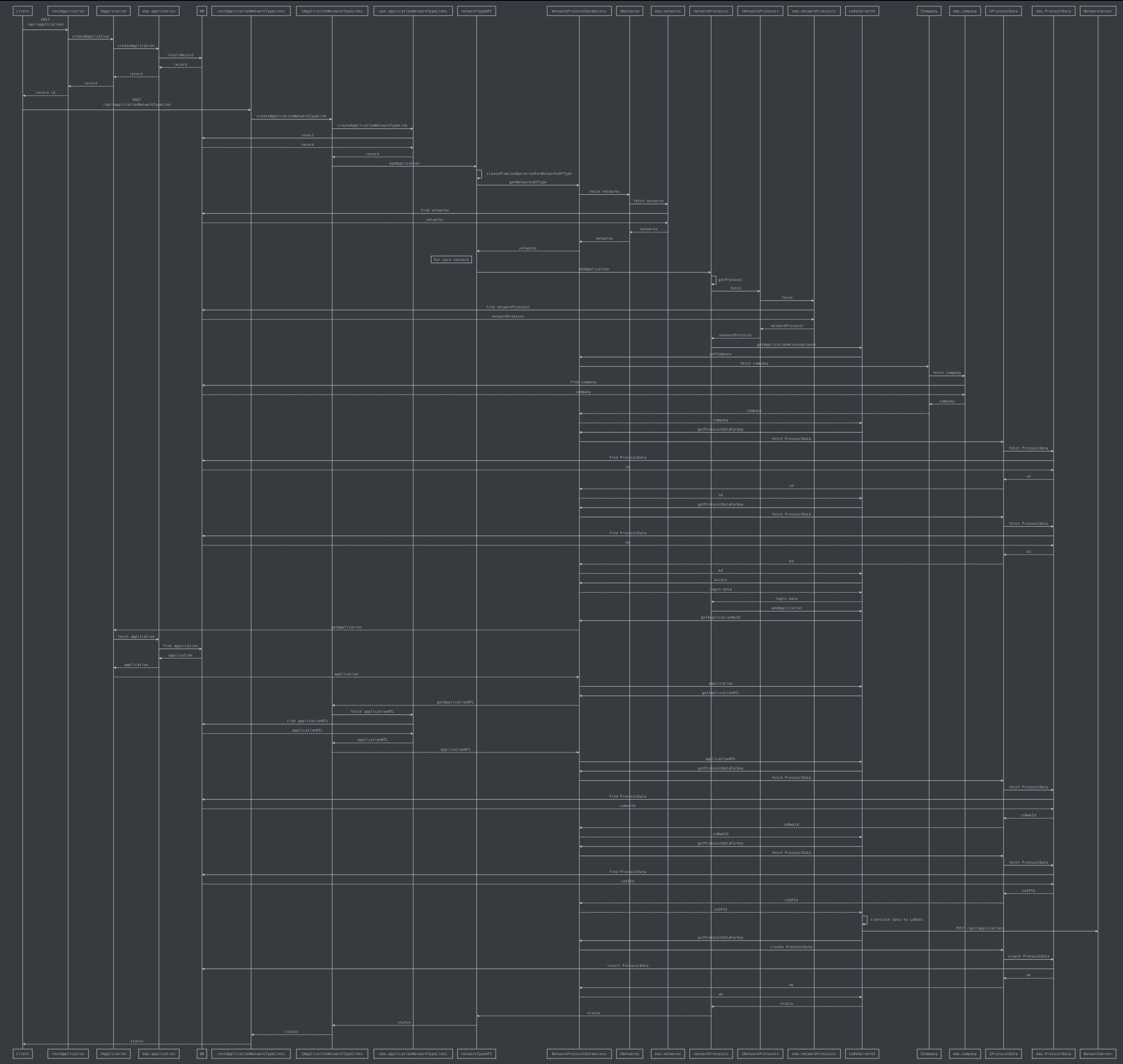


Delete Application



Current Flow

New Application


client
restApplication
IApplication
dao.application
DB
restApplicationNetworkTypeLinks
IApplicationNetworkTypeLinks
dao.applicationNetworkTypeLinks
networkTypeAPI
NetworkProtocolDataAccess
INetworks
dao.networks
networkProtocols
INetworkProtocols
dao.networkProtocols
LoRaServerV2
ICompany
dao.company
IProtocolData
dao.ProtocolData
NetworkServer
POST /api/applications
createApplication
createApplication
insertRecord
record
record
record
record.id
POST /api/applicationNetworkTypeLink
createApplicationNetworkTypeLink
createApplicationNetworkTypeLink
insert
record
record
addApplication
createPromiseOperationForNetworksOfType
getNetworksOfType
fetch networks
fetch networks
find networks
networks
networks
networks
networks
For each network
addApplication
getProtocol
fetch
fetch
find networkProtocol
networkProtocol
networkProtocol
networkProtocol
getApplicationAccessAccount
getCompany
fetch company
fetch company
find company
company
company
company
company
getProtocolDataForKey
fetch ProtocolData
fetch ProtocolData
find ProtocolData
sd
sd
sd
sd
getProtocolDataForKey
fetch ProtocolData
fetch ProtocolData
find ProtocolData
kd
kd
kd
kd
access
login data
login data
addApplication
getApplicationById
getApplication
fetch application
find application
application
application
application
application
getApplicationNTL
getApplicationNTL
fetch applicationNTL
find applicationNTL
applicationNTL
applicationNTL
applicationNTL
applicationNTL
getProtocolDataForKey
fetch ProtocolData
fetch ProtocolData
find ProtocolData
coNwkId
coNwkId
coNwkId
coNwkId
getProtocolDataForKey
fetch ProtocolData
fetch ProtocolData
find ProtocolData
coSPId
coSPId
coSPId
coSPId
translate data to LoRaV2
POST /api/applications
putProtocolDataForKey
create ProtocolData
create ProtocolData
insert ProtocolData
ok
ok
ok
status
status
status
status
status