



**Požadavky na součinnost**

**Propojení PACS - NIS**

Datum vytvoření: 17. 03. 2023



## Obsah

1. Integrace s NIS .....	3
1.1. Požadavky na dodavatele NIS .....	3
1.2. Žádanky pro modality worklist.....	3
1.3. Volání DICOM prohlížeče z prostředí NIS .....	3
1.4. Zobrazení nálezů .....	3
Příloha č. 1 - Provolání prohlížeče MARIE WebVision .....	6
Příloha č. 2 - Příklady HL7 zpráv zasílaných z NIS .....	12

## Záruky důvěrnosti

*Pokud nakládání s informacemi v tomto dokumentu nepodléhá podmínkám zvláštní písemně uzavřené smlouvy mezi společností OR-CZ spol. s r. o. a zadavatelem, budou tyto informace považovány za důvěrné a jsou určeny výhradně pro interní potřebu zadavatele. Tyto informace nesmí být bez předchozího svolení OR-CZ spol. s r. o. postoupeny třetí straně ani používány či rozmnožovány pro jiné účely, než za účelem posouzení naší nabídky.*

# 1. Integrace s NIS

Pro komplexní provoz PACS je potřeba, aby spolu komunikovaly systémy NIS a PACS

Komunikace mezi NIS a PACS probíhá v následujících úrovních:

- Žádanky pro Modality Worklist
- Volání DICOM prohlížeče z prostředí NIS
- Zobrazení nálezu z NIS v prostředí DICOM prohlížeče

Standardním komunikačním protokolem jsou HL7 zprávy. Popis formátu HL7 je uveden na stránkách [www.hl7.org](http://www.hl7.org).

Konkrétní příklady HL7 zpráv jsou uvedeny v Příloze č. 2 tohoto dokumentu.

## 1.1. Požadavky na dodavatele NIS

Objednatel s dodavatelem NIS dohodne technické parametry týkající se propojení NIS - MARIE PACS. Komunikace mezi NIS a produkty MARIE PACS jsou řešeny jak standardizovaným rozhraním HL7, tak i prostřednictvím webových služeb na straně NIS (pro zobrazení nálezu zapsaného v NIS).

## 1.2. Žádanky pro modality worklist

Při vytvoření žádanky na vyšetření v prostředí NIS je nutno tuto žádanku předat systému PACS. Po vytvoření žádanky v NIS na vyšetření zobrazovacích metod je vygenerována zpráva ve formátu HL7, která je z NIS odeslána na službu MARIE HIS Connector. Jednotlivé modality se dotazují služby MARIE HIS Connector na žádanky, které jsou určené na danou modalitu. Komunikace probíhá pomocí HL7 zprávy typu ORM^O01. Příklady zpráv jsou uvedeny v Příloze č.2.

## 1.3. Volání DICOM prohlížeče z prostředí NIS

V Příloze č.1 je uveden popis volání a spuštění webového DICOM prohlížeče z prostředí NIS. Uživatel primárně pracuje se systémem NIS a odtud má možnost volat prohlížeč pro zobrazení obrazové dokumentace daného pacienta. Volání z prostředí NIS je z hlediska dohledání konkrétního vyšetření, identifikace pacienta a uživatele, nejvhodnější variantou volání prohlížeče obrazové dokumentace.

V průběhu instalace a implementace systému předá OR-CZ dodavateli NIS všechny potřebné parametry nutné k nastavení spuštění prohlížeče MARIE WebVision z prostředí NIS. Bude se jednat o IP adresu a port serveru, na kterém je prohlížeč instalován a parametry (klíč) pro zašifrování URL adresy s parametry o vyšetření, které má být zobrazeno.

## 1.4. Zobrazení nálezů

Standardní vlastností PACS systému (DICOM prohlížeče) je možnost zobrazení nálezu k vyšetření, který byl zapsán do systému NIS. V rámci dodávaného řešení bude možno v prostředí prohlížeče MARIE WebVision zobrazit nálezy zapsané v NIS prostřednictvím volání webové služby na straně NIS, která vrátí text nálezu k danému vyšetření. Tímto je zajištěno, že je zobrazena platná verze nálezu a nedochází k nekonzistenci dat.

Nález se zobrazuje v prohlížeči MARIE WebVision v seznamu sérií jako speciální série SR (Structured Report), který je možno zobrazit kliknutím na tuto sérii.

Prohlížeč MARIE WebVision se dotazuje webové služby NIS, která vrací nález. Vstupem jsou minimálně 2 parametry (identifikace pacienta = RČ a identifikace vyšetření = číslo žádanky/accessionNumber). Je možno se z naší strany dotazovat metodou GET i POST.

### **Příklady:**

#### **1) Metoda GET**

příklad volání webové služby:

<http://localhost:8050/marie-pacs?ppId=00000000000&accessionNo=12345678>

Výstup z webové služby:

```
<?xml version="1.0" encoding="UTF-8"?>
<RequestList>
  <Request accessionNumber="12345678" studyDate="20191010" studyTime="072400">
    <Applicant icp="12345678" requestingPhysician="Žádající lékař" />
    <Examination author="Autor Popisu">
      <text>Testovací nález IKIS, MUDr. Testovací Testov
Je to zlomený:
Velmi, velmi zlomený</text>
    </Examination>
    <Patient birthdate="20191010" name="Testovací Pacient"
personalId="000000000000" sex="M" />
    <Workplace code="kód pracoviště" institution="Název Nemocnice"
shortName="Pracoviště" />
  </Request>
</RequestList>
```

#### **2) Metoda POST**

příklad volání webové služby:

<https://s-fe-app.n.cz/PACSAccessWCFService/PACSAccessWCFService.svc/RequestResultText>

Vstupní request:

```
{"UserLogon":"pacs_ad_read","AccessionNumber":"00007320222NT","PatientNationalRegNo":"6060066061"}
```

Výstup z webové služby (response):

```
{
  "Data": {
    "Author": "Testovací Lekar MUDr.",
    "LastUpdated": "/Date(1647005866180+0100)/",
```

```
"LastUpdatedDate": "2022-03-11T14:37:46",
"LastUpdatedUser": "Testovací Lekar MUDr.",
"ResultText": "RTG - Plíce vleže\r\nzaloženo: 11.03.2022 07:52 - Test Test
MUDr.\r\nposlední změna: 11.03.2022 14:37 - Testovací Lékař MUDr.\r\n\r\nText
žádanky:\r\nPac. s bilat pneumonií. Prosím kontrolní snímek vleže po OTI.
Děkuji\r\n\r\nNález:\r\nRTG plic vleže u lůžka\r\n\r\nObě plicní křídla
rozvinuta, vpravo regrese sytosti splývavého infiltrátu ve stř. poli , p.plicní
křídlo jemně závojovitě zastřené, vlevo regrese zastínění basálně, přetrvává
fluidothorax. Srdeční stín vleže levostranně rozšířen, bránice klenutá, vpravo
úhly volné. Kalcifikace Ao. Deformující spondylóza Th páteře. \r\n\r\nRES -
mírná regrese infiltrací bilat., bilat. fluidothorax\r\n\r\n\r\nPopisující
lékař: Testovací Lékař MUDr.\r\n\r\nPřístroje:\r\nTMX R+,
v.č.xxx15974sa\r\nMetody:\r\nnpsl - Prostý snímek hrudníku u
lůžka\r\nVýkony:\r\n89131 1x - RTG HRUDNÍKU\r\n\r\n"
},
"ErrorCode": 1,
"Message": null
}
```

## Příloha č. 1 - Provolání prohlížeče MARIE WebVision

### Varianta A) Volání prohlížeče v případě aktivního SSO

Tento způsob svolání je ideální v kombinaci s autentizací uživatele pomocí Single Sign-On nebo autentizací osobním certifikátem. Pokud není tento způsob autentizace povolen a uživatel není přihlášen, zobrazí se uživateli výzva k zadání uživatelského jména a hesla.

Příklady svolávací URL:

- /open?pid=123456789
- /open?accno=abc123&action=search

Seznam všech podporovaných parametrů:

- **pid** - identifikátor pacienta (např. rodné číslo)
- **pname** - jméno pacienta ve formátu DICOM Person Name (např. Novak^Jan)
- **accno** - číslo žádanky (accession number)
- **suid** - Study instance UID
- **seruid** - Series Instance UID
- **iuid** - SOP Instance UID
- **archives** - seznam AET archivů, ve který se má hledat oddělených čárkou, pokud není uvedeno, vyhledává se ve výchozích archivech pro přihlášeného uživatele
- **level** - úroveň vyhledávání, přípustné hodnoty: patient, study. Pokud není uvedeno, bere se v potaz hodnota `patient`
- **action** - akce, která se má provést
  - **open** - otevřít vyšetření (série, snímek) - výchozí hodnota, pokud parametr není uveden
  - **search** - vyhledat a zobrazit výsledky hledání
  - **fill** - vyplnit vyhledávací formulář
  - **login** - provést pouze přihlášení (použití pro API volání)

### Varianta B) Volání prohlížeče bez aktivního SSO

Tento způsob svolávání je preferovaný pro jakoukoli novou integraci systému, kde je vyžadováno předat i identitu uživatele.

Provolávání funguje na principu vytvoření tokenu, který má omezenou platnost a obsahuje informace identifikující dané vyšetření nebo pacienta a jeho obsah je digitálně podepsán algoritmem SHA512withRSA. Produkt, ze kterého se svolává, má svůj vlastní privátní klíč. Veřejný klíč je ve formátu certifikátu X.509 nahrán do instalace MARIE WebVision a označen libovolným unikátním ID. Provolávání podporuje následující parametry:

- **pid** - ID pacienta (Patient ID)
- **accno** - číslo žádanky (Accession Number)
- **suid** - Study Instance UID

- **seruid** - Series Instance UID
- **sopuid** - SOP Instance UID
- **aes** - seznam Application Entity Title, na kterých se má hledat
- **lvl** - úroveň, na které se vyhledává, podporováno: study / patient
- **user** - uživatelské jméno
- **domain** - jméno domény, do které uživatel patří
- **roles** - seznam rolí uživatele (používá se pouze v případě, že uživatel neověřuje proti LDAP ale zakládá se automaticky lokálně)
- **action** - akce, která se má provést
  - **open** - otevřít vyšetření (sérii, snímek) - výchozí hodnota, pokud parametr není uveden
  - **search** - vyhledat a zobrazit výsledky hledání
  - **fill** – vyplnit vyhledávací formulář
  - **login** - provést pouze přihlášení (použití pro API volání)
- **rnd** - náhodný řetězec nebo sekvence (pro zvýšení zabezpečení)
- **exp** - datum a čas expirace tokenu ve formátu yyyyMMddHHmmss, doporučená platnost tokenu je v řádu desítek sekund
- **key** - identifikátor RSA páru veřejného a privátního klíče
- **dataId** - identifikátor datové sady při integraci s produkty 3. stran
- **ver** - verze tokenu, 1 = podpis SHA512 RSA PKCS 1.5 padding, 2 = podpis SHA512 RSA PSS padding

Provolávající systém je zodpovědný za odstranění všech konců řádku (\n) z hodnot jednotlivých parametrů.

Jednotlivé parametry jsou sloučeny do jednoho datového řetězce ve formátu název=hodnota\n. Celý tento řetězec v kódování UTF-8 (bez BOM) je podepsán algoritmem SHA512withRSA (padding PKCS 1.5 pro verzi 1 nebo PSS pro verzi 2). Výsledný token je tvořen z:

- datové řetězce v kódování UTF-8 (bez BOM) převedeného do Base64
- symbolu pomlčka "-"
- podpisu datového řetězce převedeného do Base64

**Minimalistická implementace generátoru tokenu verze 1 v jazyce Java**

```
import java.io.FileInputStream;
import java.nio.charset.StandardCharsets;
import java.security.Key;
import java.security.KeyStore;
import java.security.PrivateKey;
import java.security.Signature;
import java.util.Base64;

public class DPGWTokenGenerator {
    public static void main(String[] args) throws Exception {

        String tokenData = "pid=1234567890\n" +
                           "user=joe.doe";

        byte[] token = tokenData.getBytes(StandardCharsets.UTF_8);
        KeyStore keyStore = KeyStore.getInstance("JKS");
        keyStore.load(new FileInputStream("keystoreFile.jks"),
                      "keystorePassword".toCharArray());

        Key key = keyStore.getKey("keyAlias",
                                  "keystorePassword".toCharArray());

        Signature signature = Signature.getInstance("SHA512withRSA");
        signature.initSign((PrivateKey) key);
        signature.update(token);
        byte[] sign = signature.sign();

        String tokenB64 = Base64.getEncoder().encodeToString(token);
        String signB64 = Base64.getEncoder().encodeToString(sign);

        System.out.println("https://dpgw/token?" + tokenB64 + '-' + signB64);
    }
}
```



**Minimalistická implementace generátoru tokenu verze 2 v jazyce Java**

```

import java.io.FileInputStream;
import java.nio.charset.StandardCharsets;
import java.security.Key;
import java.security.KeyStore;
import java.security.PrivateKey;
import java.security.Signature;
import java.util.Base64;

public class DPGWTokenGenerator {

    public static void main(String[] args) throws Exception {

        String tokenData = "pid=1234567890\n" +
                           "user=joe.doe\n"+
                           "ver=2";

        byte[] token = tokenData.getBytes(StandardCharsets.UTF_8);

        KeyStore keyStore = KeyStore.getInstance("JKS");
        keyStore.load(new FileInputStream("keystoreFile.jks"),
"keystorePassword".toCharArray());
        Key key = keyStore.getKey("keyAlias","keystorePassword".toCharArray());

        Signature signature = Signature.getInstance("RSASSA-PSS");
        signature.setParameter(new PSSParameterSpec("SHA-512", "MGF1",
MGF1ParameterSpec.SHA512, 64, 1));
        signature.initSign((PrivateKey) key);
        signature.update(token);
        byte[] sign = signature.sign();

        String tokenB64 = Base64.getEncoder().encodeToString(token);
        String signB64 = Base64.getEncoder().encodeToString(sign);

        System.out.println("https://dpgw/token?" + tokenB64 + '-' + signB64);
    }
}

```

## Minimalistická implementace generátoru tokenu verze 2 v jazyce C# .NET

```
using System;
using System.Security.Cryptography;
using System.Text;

byte[] privateKeyPem = File.ReadAllBytes(@"private-key.pem");
String baseUrl = "https://test.dicompass.cloud";
String patientId = "123";
String userName = "abc@api";

DateTime expiration = DateTime.Now.AddSeconds(30);

byte[] data = Encoding.UTF8.GetBytes("user="+userName+"\n"+
                                     "pid="+patientId+"\n"+
                                     "key=api-ae\n"+
                                     "ver=2\n"+
                                     "exp="+expiration.ToString("yyyyMMddHHmmss")
);

RSACng rsaCng = new RSACng();
rsaCng.ImportFromPem(Encoding.UTF8.GetString(privateKeyPem).ToCharArray());

byte[] signature = rsaCng.SignData(data, HashAlgorithmName.SHA512,
RSASignaturePadding.Pss);

String token = Convert.ToBase64String(data)+"-"+Convert.ToBase64String(signature);
Console.WriteLine(baseUrl+"/token?" + token);
```

## Minimalistická implementace generátoru tokenu verze 1 v jazyce PHP

```
<?php
$tokenData = "pid=123456789\nuser=joe.doe";
$pkeyid = openssl_pkey_get_private("file://private-key.pem");
openssl_sign($tokenData, $signatureData, $pkeyid, OPENSSL_ALGO_SHA512);
openssl_free_key($pkeyid);
echo "https://cloudpacs.cz/token?".base64_encode($tokenData).'-'.base64_encode($signatureData);
?>
```

## Příloha č. 2 - Příklady HL7 zpráv zasílaných z NIS

Pro manipulaci s žádankami se jedná o zprávy ORM^O01

viz: [http://hl7-definition.caristix.com:9010/HL7%20v2.3/triggerEvent/ORM\\_O01](http://hl7-definition.caristix.com:9010/HL7%20v2.3/triggerEvent/ORM_O01)

Příklady:

```
MSH|^~\&|Dicompass|Nemocnice|DPGW|DPGW|201901021300||ORM^O01|12363|P|2.3|||||8859/2
PID||123456^^^NIS|7903140003^^^RC||Einstein^Albert||18790314|M
ORC|NW|AC8003|||SC|^201901021600
OBR||AC8003|AC8003.1|PROC-001^Gastro^Medoro|||||||DIGI
```

Např. je potřebné dodat žádajícího lékaře, a to v příkladu výše není – to je vhodné do ORC.17.

```
MSH|^~\&|NIS|Nemocnice|MARIEMARIE|200409071254||ORM^O01|1402|P|2.3|||||8859/2
PID||1268223^^^NIS|KHIS001||GAMAGE^MARY^^^|19950210|F|||Ulice
cp^Mesto^PSC^CR|||||70040748
PV1||I||R|||||183494|||||200310090853
IN1||1|^1|111
AL1||1
ORC|NW|50254^NISHL7|||SC|^20040907000000^R||200409071254|||novotny^^^11211122
^2F9^333^^^NISHL7||||1121^1.INT odd. A 56789012345^NISHL7
OBR||1|50254^NISHL7|65011@50254|65011@50254^dekuji^NISHL7|||20040907000000|||||M5
45||1121^1.INT odd. A
56789012345^^^NISHL7|RTG@1|||||^20040907000000^R
```

```
MSH|^~\&|MEDSOL|MARIEMARIE|201712022316||ORM^O01|737577|P|2.3|||NE|AL|
PID|||00000558334|I^0808088888|Zkouska^Test^^^|20080808|M|Zkouska^Test^^^|Konec
na 2^Obecnice^26221^CZE^^|Konecna
2^Obecnice^26221^CZE^^|||||0808088888|||||CZE||||N
ZPI||0|
PV1||O||||val01^Jan Profesor, prof. MUDr., CSc., MBA~5020|6^72100323^^B-PK-
SC^|val01^Jan Profesor, prof. MUDr., CSc., MBA~5020|B-RDK-RTG1|||6^72100323^^B-PK-
SC^||||3242261|1|||205|||||201712022310|
ZPV|||||^72100323|^|^|
ORC|SC|3280102^MEDSOL|||||^201712022312^R||201712022311|api|von07|||||BRAD
R1EX|BRT2
OBX|1|ST|RAPEXDES^Performance descriptions^99MKW|1|RT.BRLED^ledviny
|||RT.BRLED^ledviny||F
OBX|2|ST|RAALERGI^Alergie^99MKW|ne|ne|||ne||F
OBX|3|ST|RACOMMTX^Předmět vyšetření^99MKW|1|T|||T||F
OBX|4|ST|RAEPIKRI^Epikríza^99MKW|Test 1~po odstavce 2.12.17.~Vondrka|Test 1~po
odstavce 2.12.17.~Vondrka|||Test 1~po odstavce 2.12.17.~Vondrka||F
OBX|5|ST|RAORDIAG^* Hl. diagnóza^99MKW|1|A000^Cholera, původce: Vibrio cholerae
01, bi|||A000^Cholera, puvodce: Vibrio cholerae 01, bi||F
```

```
MSH|^~\&|HIS|MedCenter|LIS|MedCenter|20060307110114||ORM^O01|MSGID20060307110114|
P|2.3
PID|||12001||Jones^John^^Mr.||19670824|M|||123 West
St.^Denver^CO^80020^USA|||||
PV1||O|OP^PAREG^|||2342^Jones^Bob|||OP|||||||2|||||||2006030
7110111|
ORC|NW|20060307110114
OBR|1|20060307110114||003038^Urinalysis^L|||20060307110114
```

Editace je pomocí ORC.1 nastaveným na XO. Zrušení je CA.

Viz odkaz výše.

#### Příklad editace patientských údajů, editace jména, příjmení, pohlaví atd. pomocí ADT.A08:

```
MSH|^~\&|REAL-ORCZ|ORCZ|DPGW|DPGW|201901021000||ADT^A08|123459|P|2.3|||||UNICODE
UTF-8
EVN|A08|201901021000
PID|1|132^^^NIS_ID|1234567890^^^RC||Novák^Jan||19850101|M|||||||
```

Zde je uveden velmi minimalistický příklad, NIS může posílat např. adresu pacienta, pro některé aplikace např. nukleární medicínu je nutná hmotnost pacienta (pro výpočet SUV), případně výška pacienta, standardně se může posílat také identifikace pojišťovny apod.

Podle standardu:

[https://hl7-definition.caristix.com/v2/HL7v2.3/TriggerEvents/ADT\\_A08](https://hl7-definition.caristix.com/v2/HL7v2.3/TriggerEvents/ADT_A08)

Při editaci ID pacienta se jedná o zprávu ADT.A40, které je v podstatě sloučení pacientů, když nový neexistuje, tak se jedná o založení nového a převěšení:

```
MSH|^~\&|REAL-ORCZ|ORCZ|DPGW|DPGW|201901021000||ADT^A40|123461|P|2.3|||||UNICODE
UTF-8
EVN|A40|201901021000
PID|1|1325^^^NIS_ID|1234567890^^^RC||Novák^Honzík||19850101|M|||||||
MRG|132^^^NIS_ID|||12345680000^^^RC|||Novak^Jan
```

[https://hl7-definition.caristix.com/v2/HL7v2.3/TriggerEvents/ADT\\_A40](https://hl7-definition.caristix.com/v2/HL7v2.3/TriggerEvents/ADT_A40)