UNIVERSITÀ DEGLI STUDI DI TRIESTE

Introduzione all’integrazione di sistemi in sanità

Parte I – Sistemi EPR/PAS/EMR – HL7 v2 – Sistemi di integrazione

Ing. Mario Damiano

TRIESTE, 11 DICEMBRE 2025

PRESENTAZIONI

ING. **MARIO DAMIANO**

LAUREA SPECIALISTICA IN INGEGNERIA CLINICA (2011)

MASTER DI II LIVELLO IN MANAGEMENT OF CLINICAL ENGINEERING (2014)

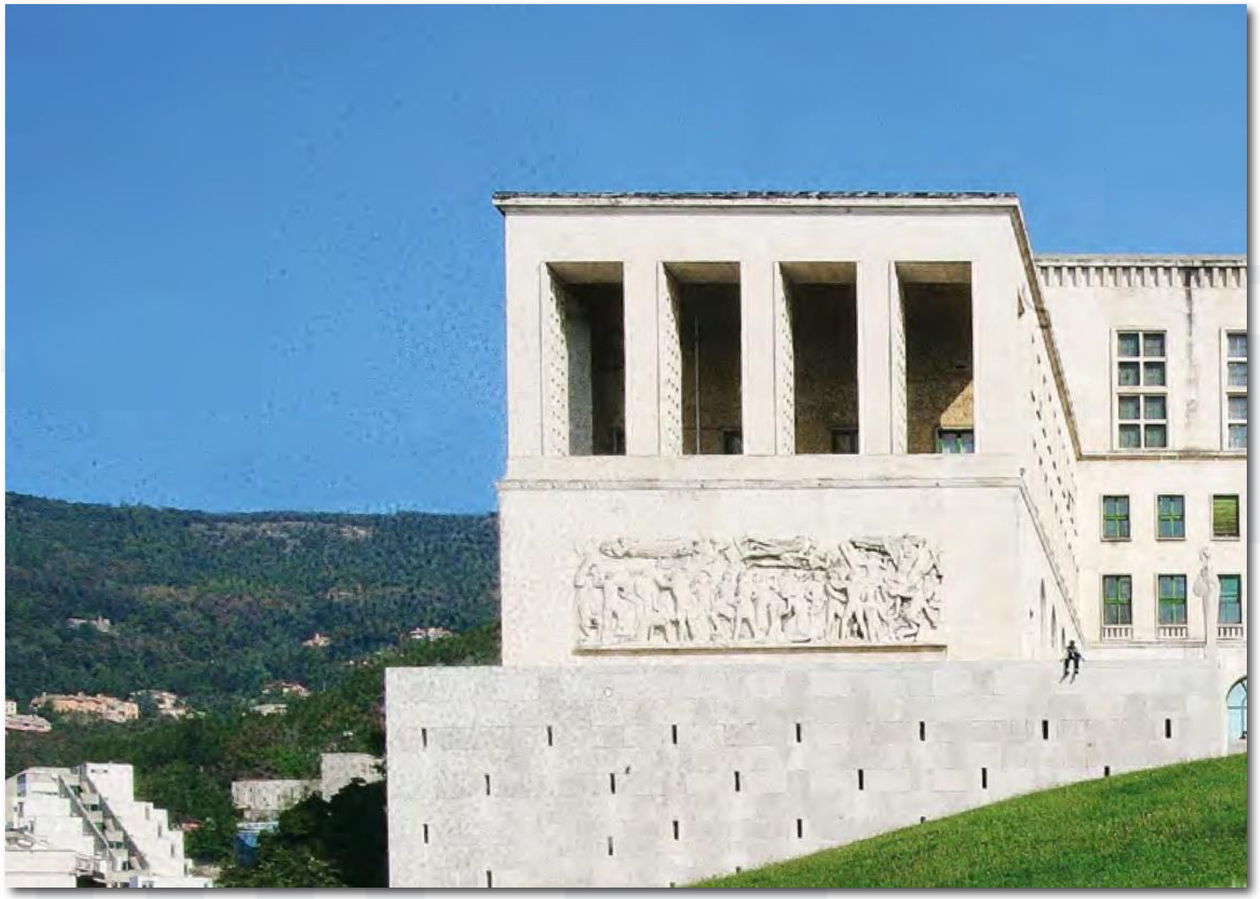
ESPERIENZA PREGRESSA NEL CAMPO DELLA MANUTENZIONE DI APPARECCHIATURE ELETTROMEDICALI CON TBS GROUP (ORA ALTHEA)

VIVO E LAVORO DAL 2014 NEL REGNO UNITO DOVE MI OCCUPO DI INTEGRAZIONE DI SISTEMI IN SANITÀ

CONTATTI:

WWW.LINKEDIN.COM/IN/MARIO-DAMIANO-26503126

EMAIL: *MARIO.DAMIANO@LIVE.COM*



AGENDA

- Sistemi informativi sanitari
- Breve cenno al sistema sanitario inglese
- Lo standard HL7 v.2
- Sistemi di integrazione
- Un esempio pratico di integrazione HL7

SISTEMI INFORMATIVI SANITARI

L’importanza della digitalizzazione dei processi

La quantità di dati clinici generata da una struttura ospedaliera è oggi giorno enorme. La spinta delle istituzioni è sempre più quella verso la digitalizzazione dei processi sanitari per consentire da un lato la gestione sicura ed efficace del dato clinico e dall’altro di migliorare l’esperienza del paziente nel suo percorso.

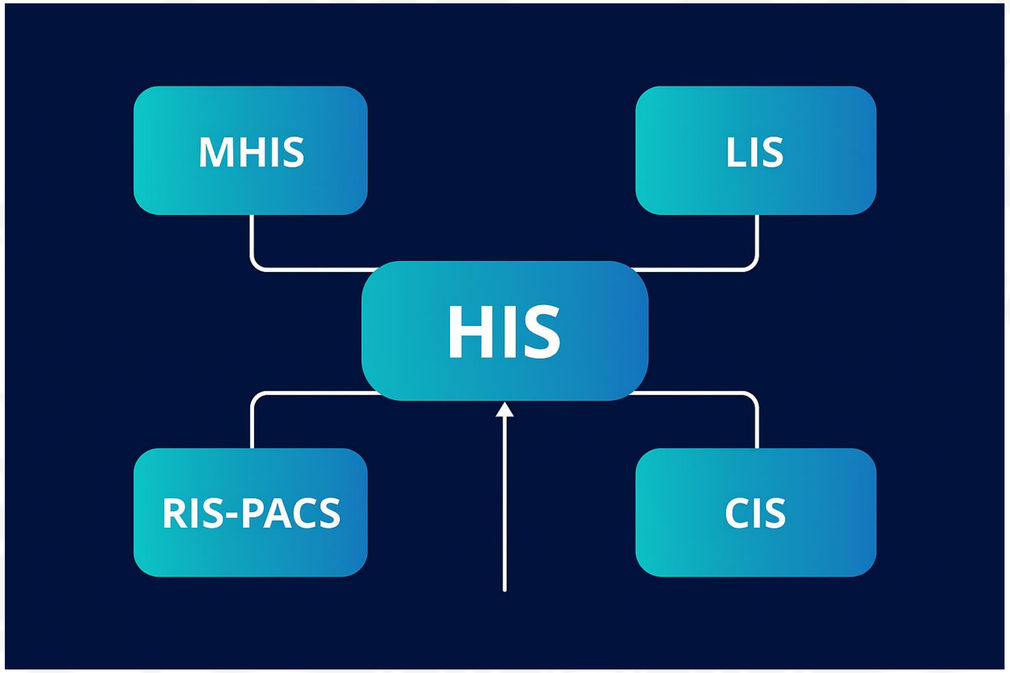
Sviluppare competenze nel mondo digitale rappresenta certamente un'opportunità per un ingegnere clinico di proiettarsi verso il futuro con possibilità nel mondo del lavoro prima inesistenti sia in Italia che all’estero.

SISTEMI INFORMATIVI SANITARI

Aspetti generali

Un sistema informativo ospedaliero (HIS) si articola in vari componenti modulari, tipicamente di *vendors* diversi in base alle esigenze cliniche e operative.

Ognuno dei componenti può afferire ad aree di competenza specifica, come ad esempio un sistema *RIS-PACS* per diagnostica clinica.



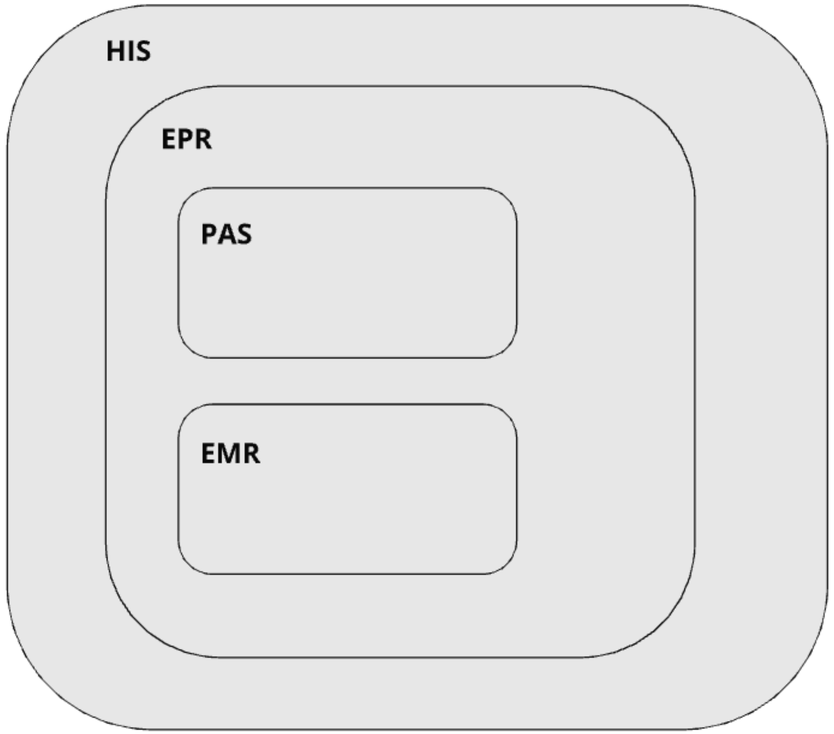
SISTEMI INFORMATIVI SANITARI

Sistemi EPR-PAS-EMR

Il cuore di un HIS è solitamente l'*electronic patient record* system (**EPR**) che ha principalmente due funzioni:

- ✓ Quella di amministrare i dati paziente, attraverso un modulo specifico detto *patient administration system* (**PAS**)
- ✓ Quella di gestire i documenti clinici associati ai pazienti, attraverso la funzione di *electronic medical record* (**EMR**)

Si tratta dunque del sistema centrale che ha l'importante compito di gestire i dati paziente nel suo complesso garantendo integrità e unicità del dato.



SISTEMI INFORMATIVI SANITARI

Principali vendors di sistemi EPR sul mercato italiano ed internazionale

Epic

Cerner

SIEMENS Healthineers

MEDITECH

AGFA HealthCare

InterSystems TrakCare

Dedalus

PHILIPS Healthcare

gpi The Healthcare Partner

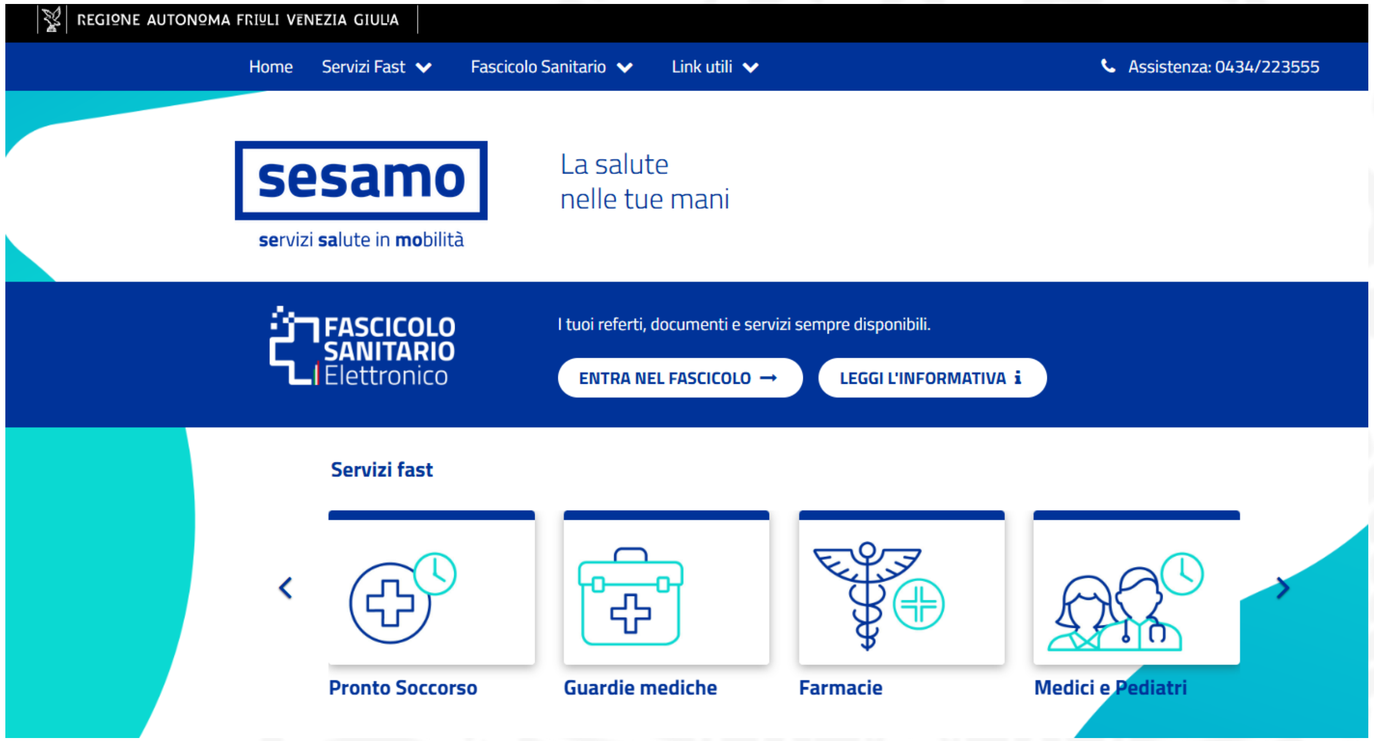
SISTEMI INFORMATIVI SANITARI

Sistemi EPR-PAS-EMR

Tali sistemi hanno una elevata scalabilità. La tendenza negli ultimi anni è quella di raggruppare coorti di pazienti sempre più grandi su base territoriale.

La Regione Friuli Venezia Giulia, come già molte altre in Italia, implementa il *fascicolo sanitario elettronico* (**FSE**) dove sono inseriti documenti sanitari di vario tipo, quali referti di diagnostica, prescrizioni di farmaci, lettere di dimissioni, vaccinazioni.

Il sistema è inoltre integrato con l'*Anagrafe Nazionale degli Assistiti* (**ANA**).

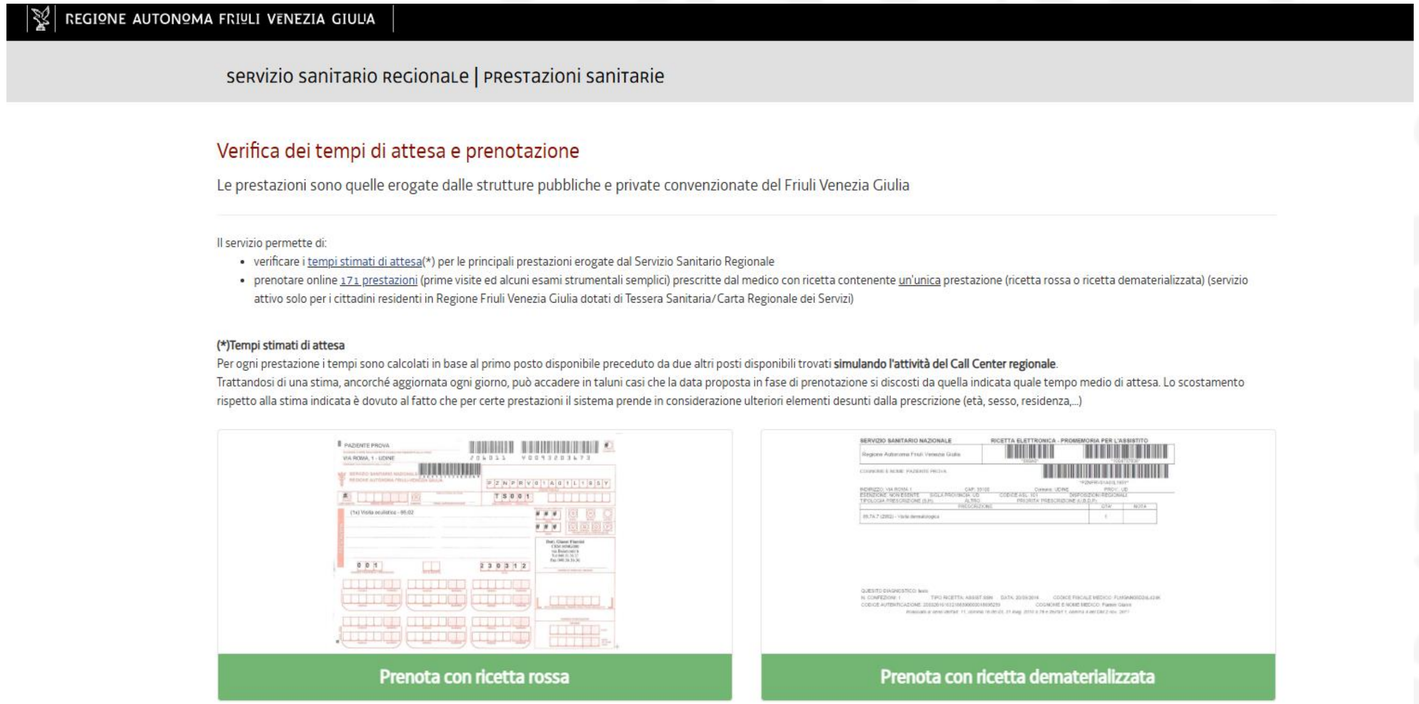


SISTEMI INFORMATIVI SANITARI

Sistemi EPR-PAS-EMR

Tali sistemi possono inoltre dare ai pazienti la possibilità di prenotare direttamente i loro appuntamenti se in possesso di impegnativa del medico (ricetta rossa o digitale).

Il CUP regionale puo inoltre gestire le prenotazioni telefonicamente utilizzando un applicativo (SDAWeb)



SISTEMI INFORMATIVI SANITARI

Criticità

Essendo i vari componenti di un sistema informativo ospedaliero prodotti da *vendor* diversi, si e sempre presentata l'esigenza di far comunicare sistemi diversi in modo efficiente e sicuro.

Si presenta dunque l'esigenza di raggiungere un'adeguata **interoperabilità** tra diversi sistemi. Secondo la definizione della *Healthcare Information and Management Systems Society* (**HIMSS**)

> *L'interoperabilità sanitaria è la capacità di diversi sistemi, dispositivi e applicazioni sanitari di accedere, scambiare, integrare e utilizzare i dati senza problemi.*

L'utilizzazione di standard di comunicazione e di workflow predefiniti permette dunque oggi di raggiungere elevati livelli di interoperabilità tra sistemi.

SISTEMI INFORMATIVI SANITARI

Il flusso dati

Per gestire l'enorme mole di dati scambiati tra i vari sottosistemi, con azione coordinante ad opera del sistema centrale, è necessario dunque l'utilizzo di standard di dati e di comunicazione predefiniti.

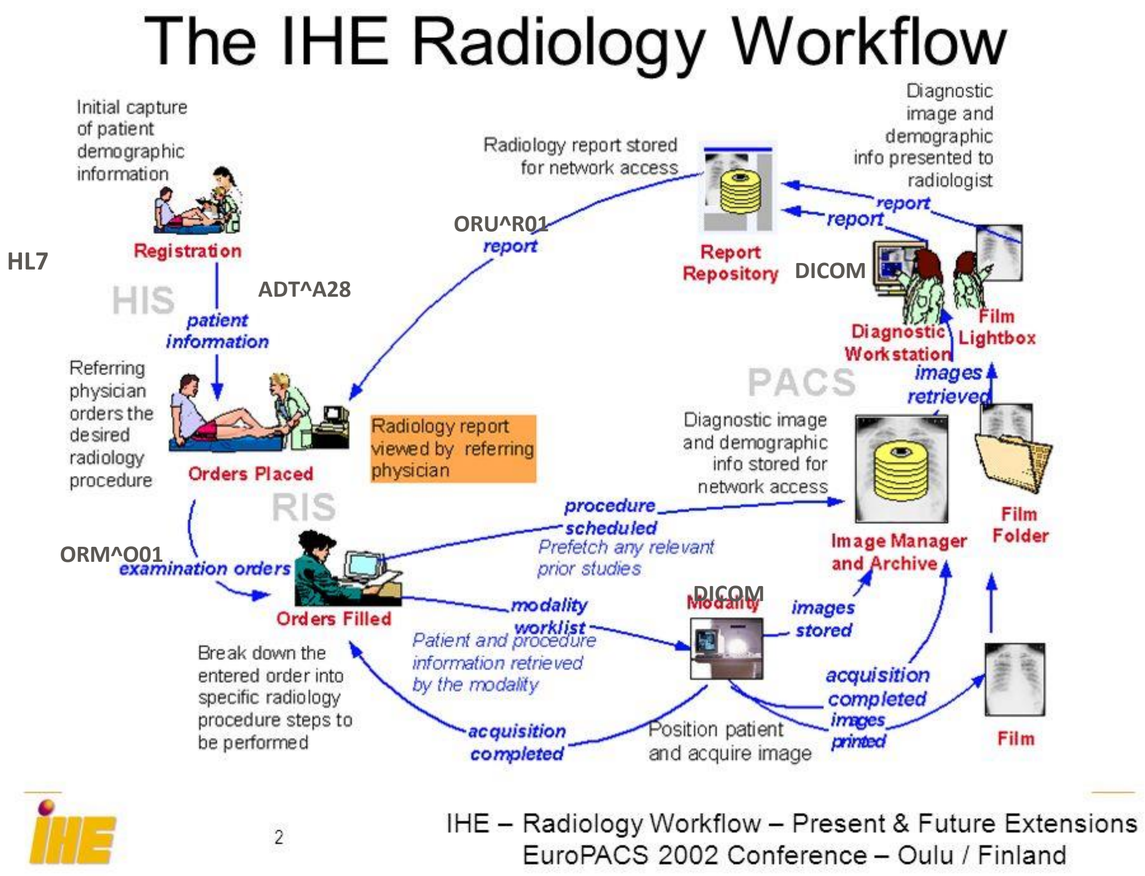
Integrating the Healthcare Enterprise (**IHE**) è un'organizzazione internazionale no-profit con sede negli Stati Uniti che lavora sia con i principali produttori di sistemi informativi sanitari che con le principali associazioni internazionali di settore per promuovere l'utilizzo efficace degli standard al fine di condividere dati ed informazioni.

IHE propone una serie di technical frameworks come modelli di integrazione, oggi largamente utilizzati nell'industria.

https://www.ihe.net/

SISTEMI INFORMATIVI SANITARI

Radiology - Scheduled Workflow



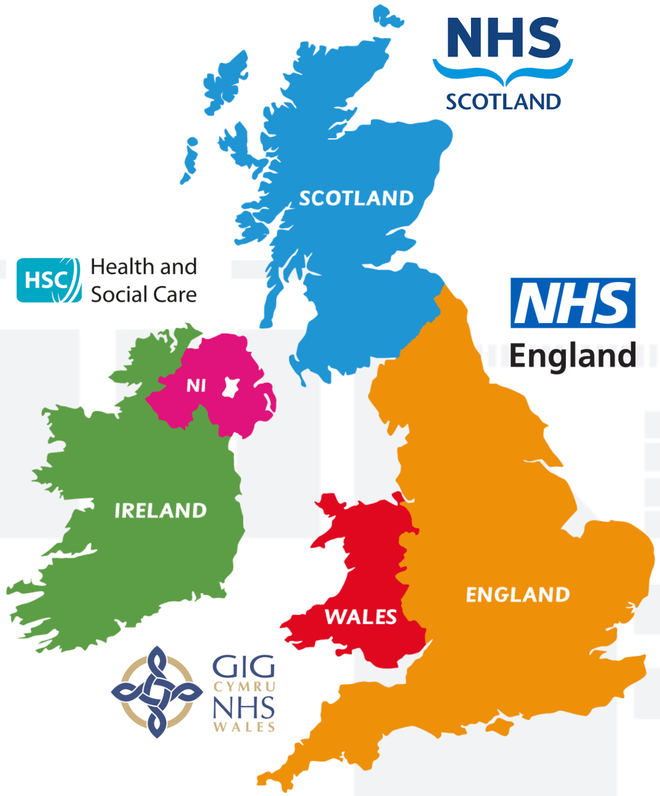
Esempio di un profilo di integrazione in radiologia (*scheduled workflow*).

Il profilo definisce attori ed azioni e suggerisce lo standard di comunicazione da utilizzare, in questo caso HL7 per la parte di anagrafica e gestione dell'ordine e DICOM per la gestione delle immagini diagnostiche.

Si noti come la registrazione del paziente non debba necessariamente avvenire localmente in quanto tali dati possono essere forniti da sistemi centralizzati, come appunto l'Anagrafe Nazionale degli Assistiti in Italia.

IL SISTEMA SANITARIO NEL REGNO UNITO

Aspetti generali



A differenza dell'Italia, dove esiste un unico sistema sanitario con autonomia amministrativa su base regionale, il sistema sanitario britannico (*National Health Service*) di carattere pubblico e in gran parte gratuito si articola in quattro distinti sistemi, uno per ciascuna delle nazioni costitutive del Regno Unito.

NHS England eroga servizi sanitari in Inghilterra ed è il più grande dei quattro. Ogni giorno **1.3 milioni** di pazienti vengono visitati, pari alla popolazione dell'Estonia. Si articola in **229** trusts, a loro volta suddivisi in:

Acute trusts (equivalenti alle nostre aziende ospedaliere)

Community services trusts (equivalenti alle nostre aziende sanitarie locali)

Mental health trusts (community services e ospedali con target specifico sul mental health)

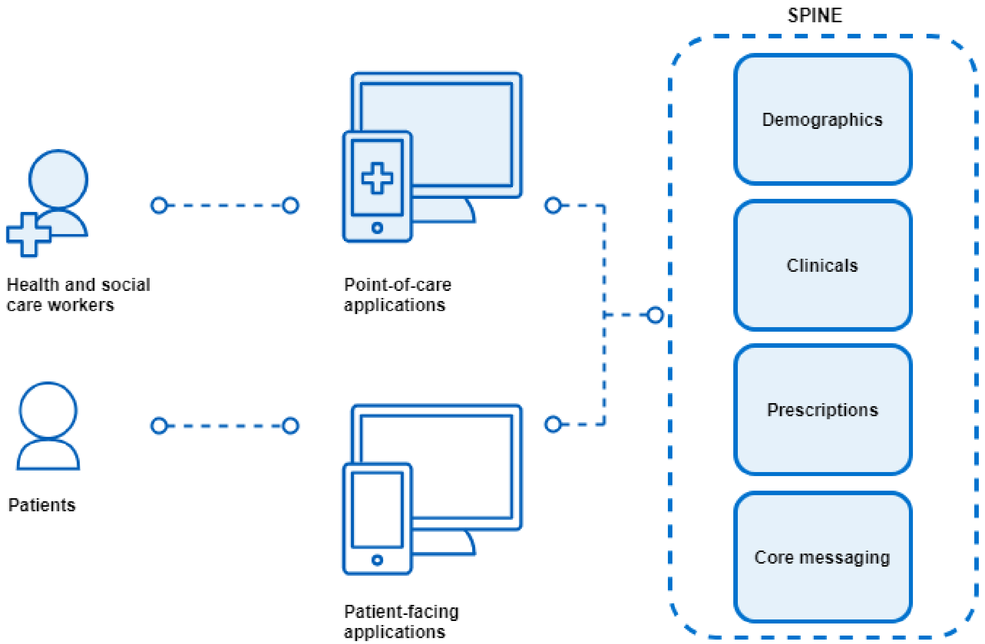
Inoltre, sono presenti circa 7000 cliniche di medici di medicina generale, chiamate GP surgeries.

A differenza di quanto accade attualmente in Italia, la gestione dei dati clinici e anagrafici in UK è più frammentata, dove i singoli acute trusts tendono a gestire i dati in piena indipendenza.

GP surgeries e community services usano invece un sistema centralizzato (chiamato *SPINE*)

IL SISTEMA SANITARIO NEL REGNO UNITO

Un esempio di sistema informativo nazionale



Spine è utilizzato per gestire centralmente a livello nazionale in Inghilterra (*NHS England*) i dati clinici e anagrafici nonché le prescrizioni di farmaci.

I dati possono essere inseriti o visionati da personale sanitario o anche dai pazienti stessi attraverso app dedicate.

Alcuni vendors offrono soluzioni per GP surgeries dove i pazienti possono richiedere digitalmente un appuntamento e visionare i propri dati.

Ultimamente in seguito alla spinta verso la digitalizzazione dei servizi sanitari in seguito all'epidemia di COVID 19, è stata introdotta un'app a livello nazionale che integra non solo i dati di GP surgeries (*primary care*) e community services ma anche gli appuntamenti specialistici erogati da ospedali acuti (*secondary care*).

NHS App

UNIVERSITÀ
DEGLI STUDI
DI TRIESTE

LO STANDARD HL7 v.2

LO STANDARD HL7 V.2

INTRODUZIONE

HL7 (*Health Level 7*) v.2 nasce alla fine degli anni 80 dall'esigenza di dover uniformare e standardizzare il processo di scambio di informazioni tra sistemi sanitari.

Si articola in diverse sotto versioni dalla v.2.2 alla più recente v.2.9.

Ogni paese può inoltre introdurre un più elevato livello di customizzazione in base alle esigenze. Nel Regno Unito ad esempio esiste un gruppo di lavoro noto come HL7 UK che ha la funzione di interagire con l'organizzazione internazionale HL7 al fine di assicurarsi che lo standard sia efficacemente utilizzato all'interno del sistema sanitario britannico.

Esiste una versione completamente nuova, la v.3, basata sul formato XML ma scarsamente utilizzata nell'industria a causa dell'introduzione di un formato più moderno e funzionale (FHIR).

https://www.hl7.org.uk/
https://www.hl7.it/
https://www.hl7.org/

LO STANDARD HL7 V.2

Struttura di un messaggio HL7 (1)

HL7 è *event based* ovvero ad ogni azione nell'interfaccia corrisponde la generazione di un messaggio che viene inviato verso il sistema ricevente.

I messaggi permettono dunque la comunicazione p2p in un'ottica di interoperabilità tra sistemi.

I messaggi sono facili da interpretare a prima vista grazie alla semplicita con cui sono strutturati.

Esempio di un messaggio ADT^A28 (registrazione di un nuovo paziente):

MSH|^~\&|TRUST_PAS|TRUST|SCHEDULING|EMR|20190822140527||ADT^A28|0000000000820327|D|2.4|||AL|NE|GBR||||v3_5_3

EVN|A28|20190822140527|||||

PID|||0123123^^^^MRN~0000255355^^^^NHS||JONES^JOAN^^^MISS||19990521|2|||10 SOME STREET^^TESTTOWN^^SY8 1NP^^HOME||02111111111^^^HOME^^^^^Y~07999123123^^^MOBILE^^^^^N~test@test.com^^^EMAIL^^^^^Y|||||||||||||||20191212|

LO STANDARD HL7 V.2

Struttura di un messaggio HL7 (2)

Il messaggio è strutturato in una serie di segmenti, ciascuno dei quali identificato da un codice alfanumerico standardizzato. Nell'esempio riportato, il messaggio è suddiviso in tre segmenti:

Segmento **MSH** (*message header*)

MSH|^~\&|TRUST_PAS|TRUST|SCHEDULING|EMR|20190822140527||ADT^A28|0000000000820327|D|2.4|||AL|NE|GBR||||v3_5_3

Segmento **EVN** (*event*)

EVN|A28|20190822140527|||||

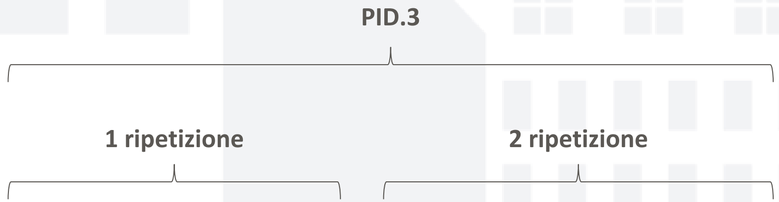
Segmento **PID** (*patient identifier*)

PID|||0123123^^^^MRN~0000255355^^^^NHS||JONES^JOAN^^^MISS||19990521|2|||10 SOME STREET^^TESTTOWN^^SY8 1NP^^HOME||02111111111^^^HOME^^^^^Y~07999123123^^^MOBILE^^^^^N~test@test.com^^^EMAIL^^^^^Y||||||||||||||||20191212|

LO STANDARD HL7 V.2

Struttura di un segmento

Andiamo a prendere l'esempio del segmento PID (*patient identification*). Per determinare la posizione del campo, si conta il numero dei campi a partire dal codice alfanumerico come una sorta di indirizzo. Tali campi sono separati da un vertical pipe -> | , i sottocampi sono separati da caret -> ^ mentre le ripetizioni dei campi da tilde -> ~. PID.3.1 ad esempio corrisponde all'identificativo del paziente **0123123** nella prima ripetizione di PID.3, dove PID.3.5 contiene il valore «*MRN*». Nella seconda ripetizione, PID.3.1 contiene il valore **0000255355**, dove il contenuto di PID.3.5 è «*NHS*». Successive ripetizioni del campo PID.3 permettono dunque in maniera efficace di gestire identificativi multipli in modo strutturato.

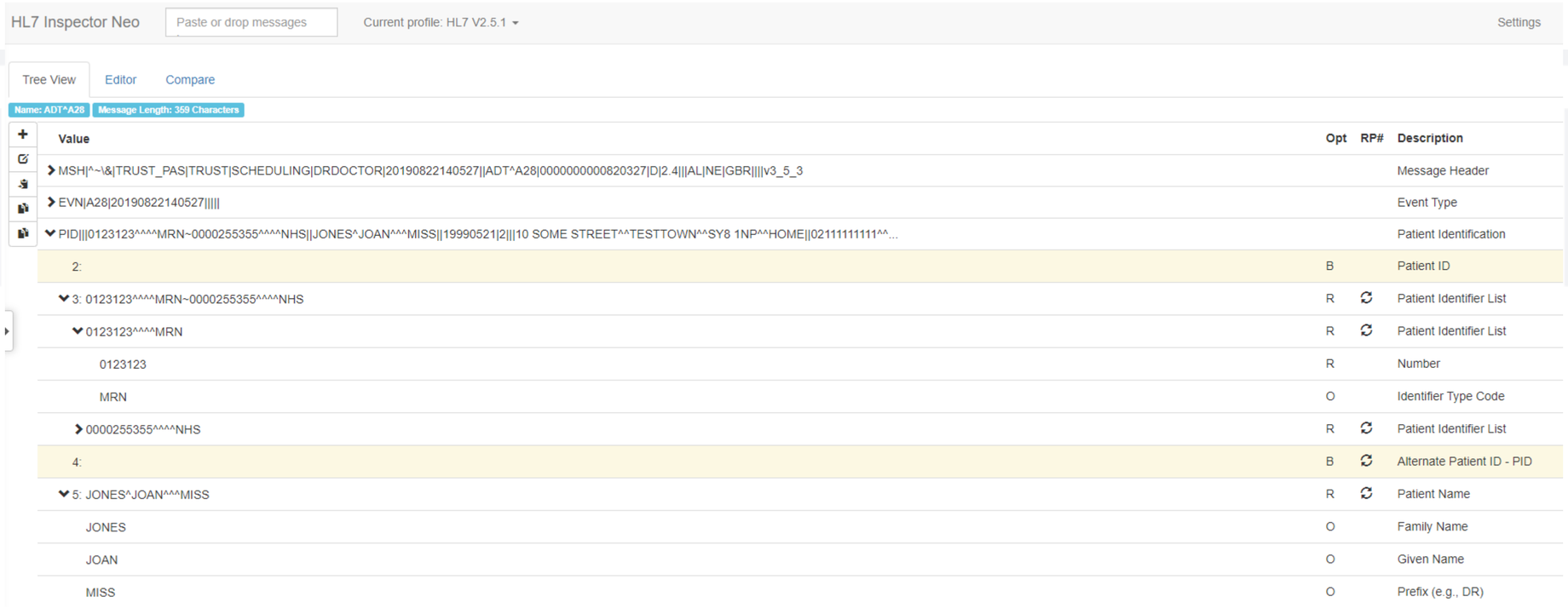


PID|||***0123123***^^^^***MRN***~***0000255355***^^^^***NHS***||*JONES^JOAN^^^MISS||19990521|2|||10 SOME STREET^^TESTTOWN^^SY8 1NP^^HOME||02111111111^^^HOME^^^^^Y~07999123123^^^MOBILE^^^^^N~test@test.com^^^EMAIL^^^^^Y||||||||||||||20191212|*

LO STANDARD HL7 V.2

HL7 inspector

Esistono vari HL7 viewers online come ad esempio *HL7 inspector* (https://www.hl7inspector.com) che permettono con grande facilità di analizzare un messaggio HL7 identificando i principali componenti e le informazioni contenute. I nomi dei campi sono definiti dallo standard e indicati come richiesti oppure opzionali.



LO STANDARD HL7 V.2

Esempi di tipologie di messaggi

Anagrafica

ADT^A28 (nuovo paziente)
ADT^A31 (update)
ADT^A40 (merge)

Ordini

ORM^O01 (nuovo ordine)
ORU^R01 (report)

Scheduling

SIU^S12 (nuovo appuntamento)
SIU^S15 (cancellazione appuntamento)

ADT

ADT^A01 (nuova ammissione)
ADT^A02 (trasferimento)
ADT^A03 (dimissione)

Laboratorio

OLM^O21 (nuovo ordine di laboratorio)
OLM^O33 (ordine multiplo per singolo campione)

LO STANDARD HL7 V.2

Come vengono inviati i messaggi

La maggior parte dei sistemi con funzionalità HL7 solitamente permette di inviare o ricevere messaggi HL7 attraverso vari protocolli (TCP/HTTP/FTP).

I principali limiti tipicamente sono:

1) Pur aderendo allo standard HL7, i messaggi sono solitamente in un formato precompilato che non può essere modificato. Ciò pone un limite se il sistema ricevente necessita di alcuni campi specifici
2) E' difficile se non impossibile controllare il flusso dati correttamente, ovvero filtrare messaggi sulla base della loro tipologia e di quali sistemi debbano ricevere
3) Gestione degli errori non ottimale, non si ha un quadro completo dello scambio di dati con rischio concreto di data loss
4) Impossibilità di gestire code di input ovvero di poter accodare temporaneamente i messaggi in arrivo per evitare perdite, specialmente quando il throughput del sistema ricevente è inferiore al ritmo di arrivo dei messaggi
5) Misure di protezione dei dati come criptazione SSL spesso non disponibili

UNIVERSITÀ
DEGLI STUDI
DI TRIESTE

SISTEMI DI INTEGRAZIONE

SISTEMI DI INTEGRAZIONE

Aspetti generali

La soluzione a tali problemi è quella di utilizzare un sistema di integrazione (*integration engine*) che permetta di collegare efficacemente sistemi sorgente ai rispettivi sistemi riceventi e viceversa in modo efficace e sicuro.

Un sistema di integrazione è un middleware che si interpone tra il sistema inviante e più sistemi riceventi con le seguenti funzioni principali:

- ✓**Ricezione dei dati**: accettare messaggi da diverse fonti (EHR, LIS, RIS) attraverso vari protocolli di trasmissione (TCP/IP, HTTP, FTP).
- ✓**Normalizzazione dei dati**: convertire formati diversi (HL7, XML, JSON) in uno standard comune per facilitare l'elaborazione.
- ✓**Trasformazione dei dati**: applicare logiche di trasformazione sui dati in entrata per conformarli alle esigenze dei sistemi destinatari, utilizzando sia linguaggi di programmazione che comandi e funzioni nella UI.
- ✓**Validazione dei dati**: certificare che i dati ricevuti rispettino le regole del formato (es. HL7) e correggere eventuali errori.
- ✓**Routing dei messaggi**: determinare la destinazione dei messaggi in base a criteri configurati (tipo di messaggio, contenuto, regole aziendali).

SISTEMI DI INTEGRAZIONE

Aspetti generali

- ✓**Riconciliazione e correlazione**: associare messaggi correlati (ad es. report di esami radiologici ai rispettivi ordini) per garantire la coerenza nei flussi di lavoro.
- ✓**Gestione delle code**: accodare messaggi per garantire la gestione efficace dei carichi di lavoro, evitando perdite di dati in caso di errori.
- ✓**Monitoraggio e logging**: tracciare lo stato dei messaggi, con la possibilità di monitorare, loggare e notificare errori o successi.
- ✓**Retry ed error handling**: implementare meccanismi di retry per i messaggi in stato di errore e gestire gli errori in modo automatico o manuale.
- ✓**Sicurezza dei dati**: garantire la cifratura, autenticazione e autorizzazione per la protezione dei dati sensibili (ad es. tramite SSL/TLS o crittografia dei messaggi).
- ✓**Scalabilità e bilanciamento del carico**: scalare il sistema per gestire aumenti di traffico o garantire l'alta disponibilità attraverso il bilanciamento del carico.

SISTEMI DI INTEGRAZIONE

Principali sistemi di integrazione

I principali sistemi di integrazione disponibili sul mercato s comunemente usati nell'industria sono solitamente:

Rhapsody: noto per la sua affidabilità, scalabilità e facilità d'uso. Supporta vari standard di messaggistica sanitaria come HL7, FHIR, DICOM e X12, ed è ampiamente utilizzato per la sua flessibilità nella configurazione e gestione delle interfacce.

Ensemble: parte della piattaforma InterSystems IRIS, Ensemble è una soluzione completa che combina integrazione, gestione dei dati e sviluppo di applicazioni. Si distingue per la sua capacità di gestire grandi volumi di dati in tempo reale, grazie alla sua potente architettura di database.

Mirth Connect: prodotto open source (oggi parte di NextGen Healthcare), Mirth Connect è apprezzato per la sua versatilità e il suo costo accessibile. Supporta diversi standard di messaggistica (come HL7, DICOM, FHIR) e offre strumenti per monitoraggio, trasformazione e instradamento dei messaggi in ambienti sanitari.

Cloverleaf: parte della suite Infor, Cloverleaf è un motore di integrazione robusto che offre strumenti avanzati per la gestione dei flussi di dati complessi tra sistemi sanitari. È altamente personalizzabile e supporta un'ampia gamma di standard di interoperabilità, come HL7, FHIR, e X12.

SISTEMI DI INTEGRAZIONE

Mirth Connect

Mirth Connect è oggi uno dei sistemi di integrazione più diffusi, se non il più diffuso in assoluto, per via del fatto che è una soluzione open source che permette di essere utilizzata efficacemente con grande scalabilità da piccole realtà come cliniche ambulatoriali private fino a centri ospedalieri di carattere regionale e nazionale.

Mirth è basato su una piattaforma Java e utilizza JavaScript come linguaggio di programmazione. Necessita di un database per funzionare, il database di default è Apache Derby che va bene solo per scopi dimostrativi. In generale un'installazione di Mirth Connect è quasi sempre abbinata a un database Microsoft SQL

Si può scaricare gratuitamente dal sito di NextGen:

https://www.nextgen.com/solutions/interoperability/mirth-integration-engine/mirth-connect-downloads

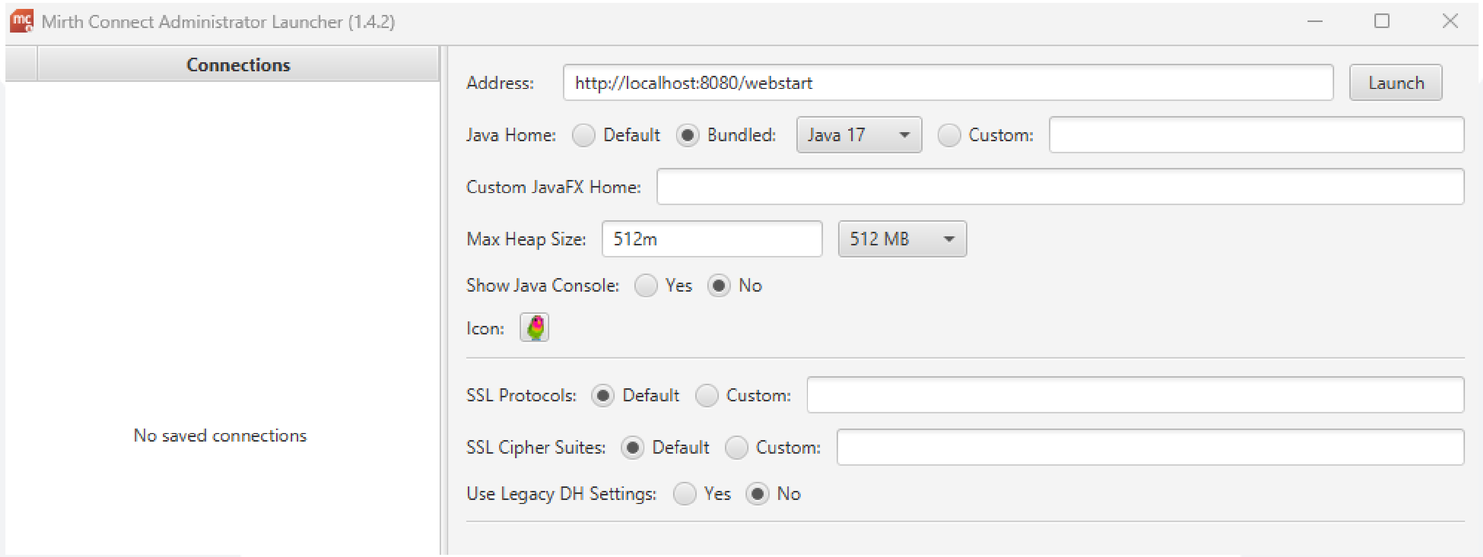
Per chi ha familiarità con T-SQL, si può anche abbinare a SQL Express, gratuitamente scaricabile dal sito di Microsoft:

https://www.microsoft.com/en-gb/sql-server/sql-server-downloads

SISTEMI DI INTEGRAZIONE

Mirth Connect

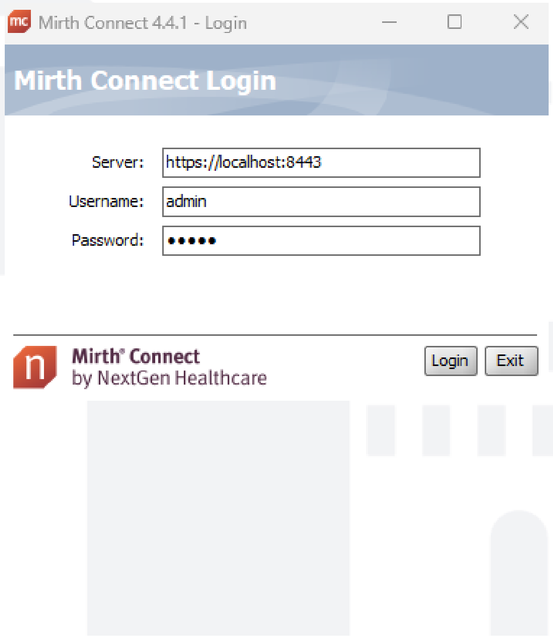
Una volta installato, Mirth è accessibile attraverso il *Mirth Admistrator Launcher*. Si noti come Mirth viene installato come un web service che può essere installato localmente oppure su un server dedicato.

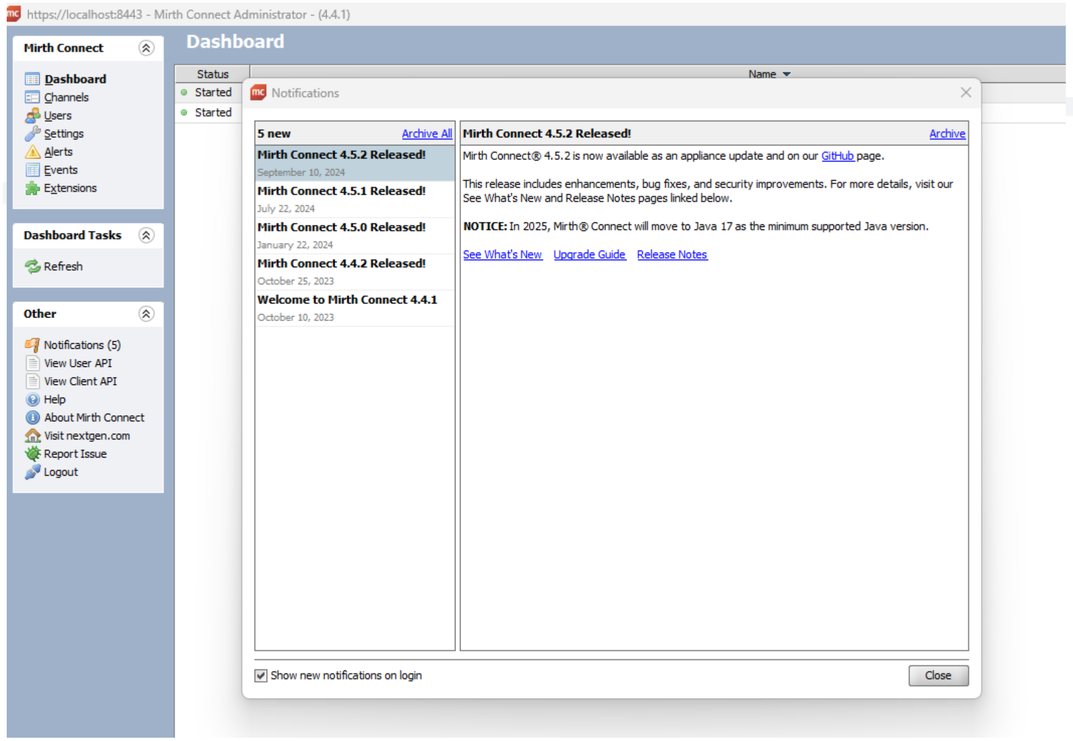


SISTEMI DI INTEGRAZIONE

Mirth Connect

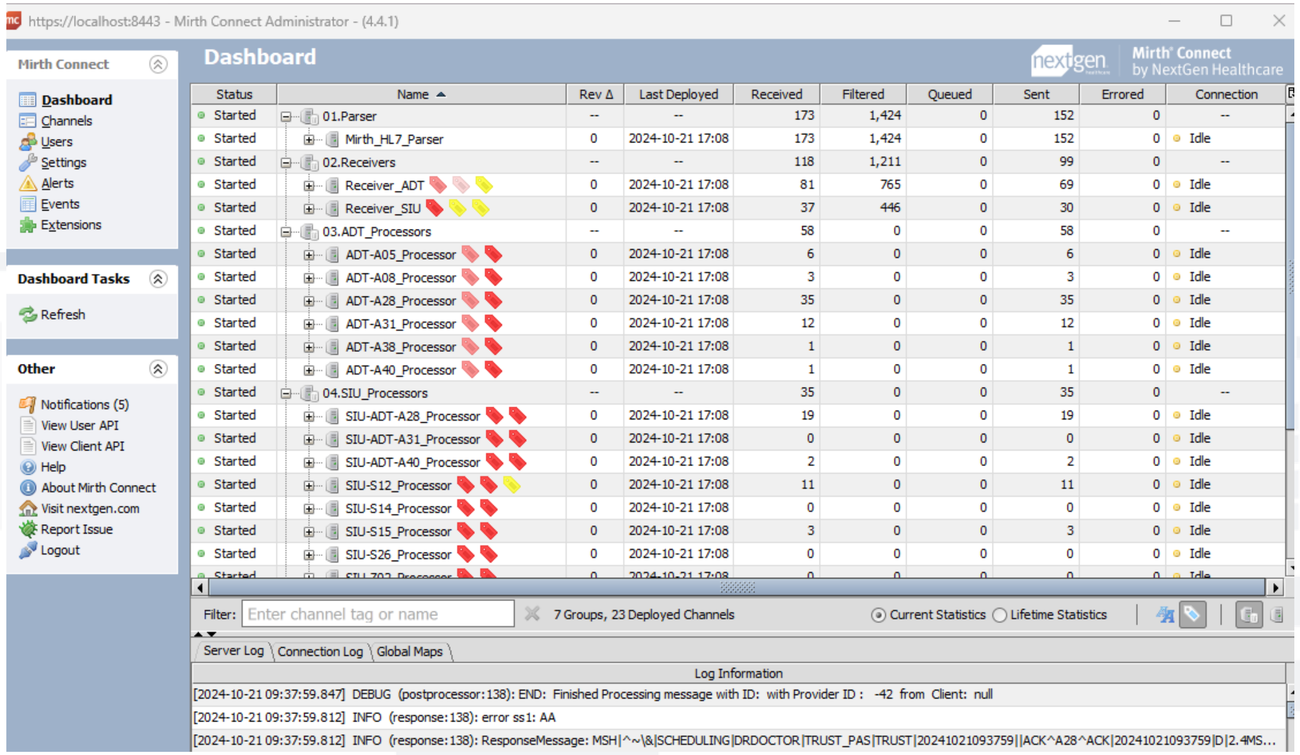
Una volta caricate le risorse Java, si viene invitati a inserire username e password, dopodiché si accede alla schermata principale: il *Dashboard* screen.





SISTEMI DI INTEGRAZIONE

Mirth Connect – Dashboard Screen

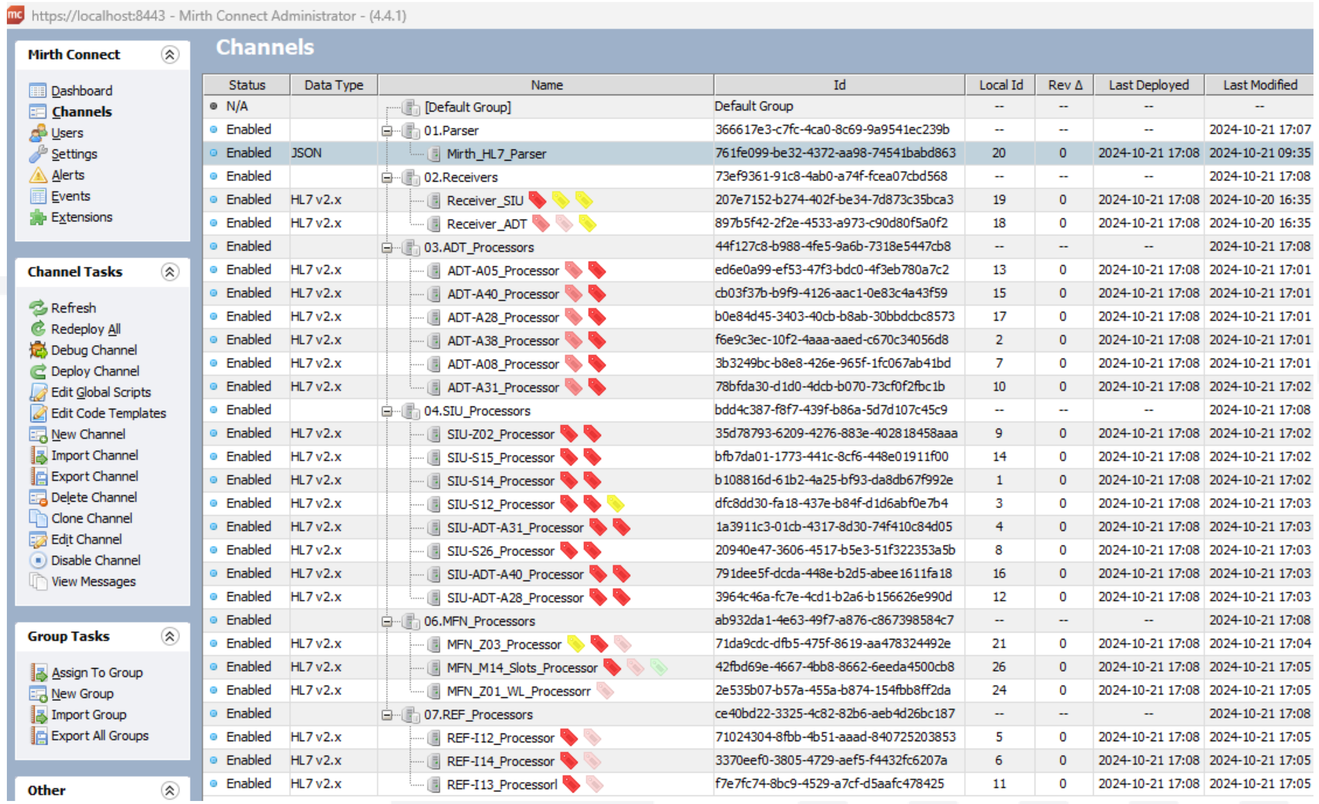


Il dashboard permette di tenere sotto controllo i singoli canali (*channels*) con una overview completa sui volumi di traffico, ovvero messaggi ricevuti, inviati, filtrati, in coda e in stato di errore.

Mirth è un middleware che può essere sostanzialmente schematizzato come un sistema ingresso/uscita, dove il messaggio ricevuto può essere filtrato e trasformato prima di essere inviato ai sistemi downstream

SISTEMI DI INTEGRAZIONE

Mirth Connect – Channel Screen



Il Channels screen permette invece di creare o modificare i vari channels con tutta una serie di funzioni a disposizione nel Channel Tasks menu.

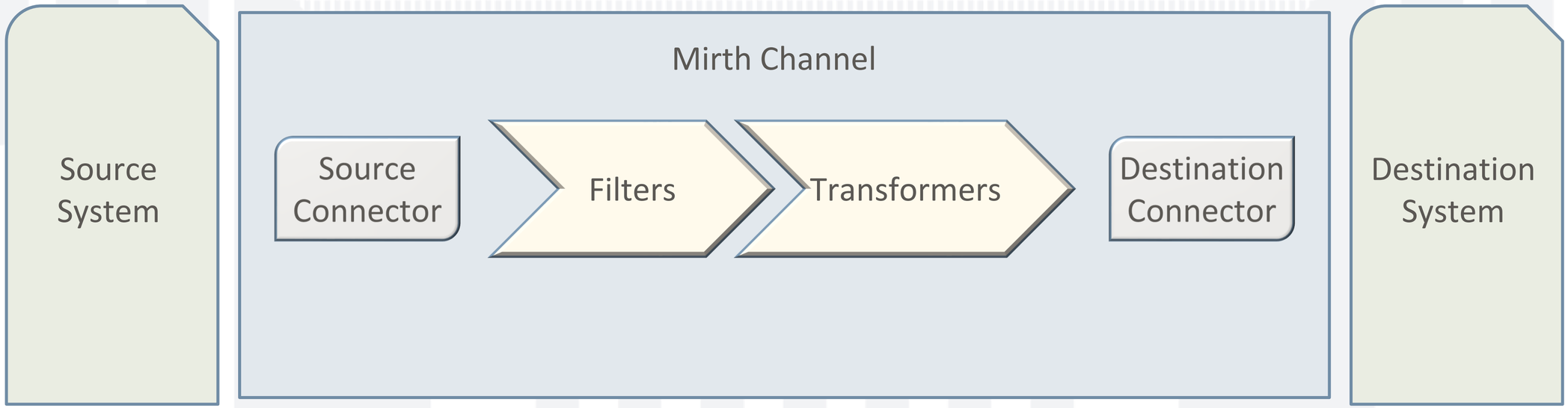
I channels una volta creati possono essere utilizzati (deployed) e quindi accessibili nel dashboard screen.

SISTEMI DI INTEGRAZIONE

Struttura di un channel

L'unita elementare di un interfaccia di integrazione in Mirth Connect è il *channel*. E' composto da due componenti principali: un *source connector*, che riceve i dati da un sistema sorgente, e da uno o più *destination connector* che hanno il compito di trasmettere dati ai sistemi riceventi.

Nel channel è possibile creare di trasformazioni o filtri sia utilizzando il linguaggio nativo di Mirth, *JavaScript*, oppure attraverso gli strumenti a disposizione nell'interfaccia utente.



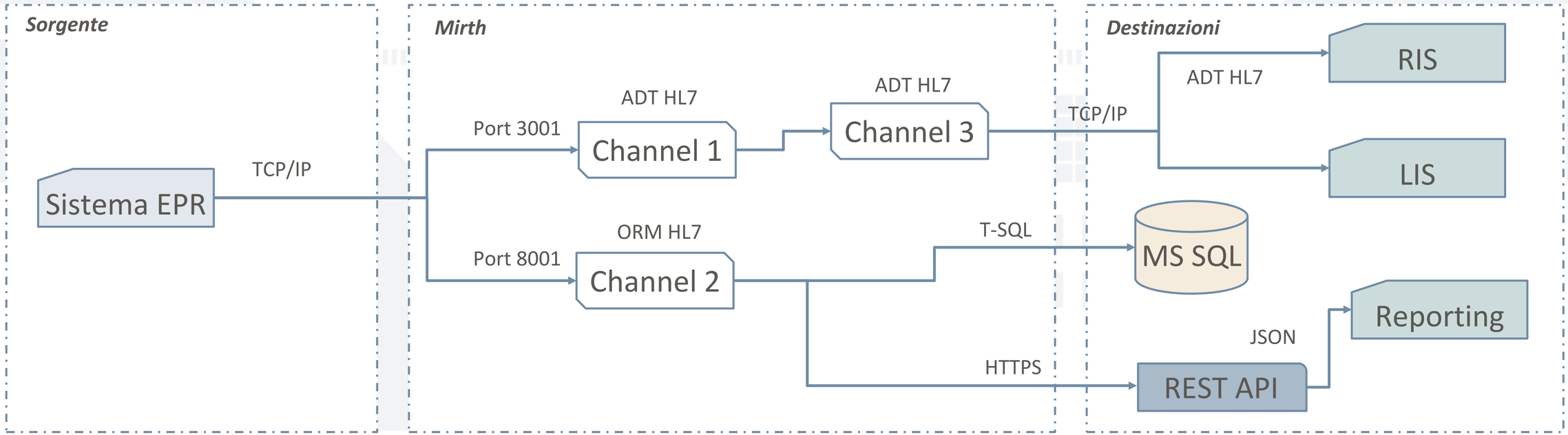
SISTEMI DI INTEGRAZIONE

Schema di un interfaccia con più channels

Nell'esempio proposto si può osservare un sistema EPR sorgente che fornisce due flussi dati (anagrafica e ordini) utilizzando messaggi HL7 di tipo ADT e ORM e attraverso due connessioni TCP/IP separate verso due channels che agiscono da TCP listener.

Channel 1 invia dopo alcune trasformazioni al Channel 3 che alimenta tramite ulteriori connessioni TCP/IP i due sistemi riceventi di radiologia e laboratorio.

Nel frattempo, il Channel 2 trasforma gli ordini in modo da poter alimentare un database di tipo MS SQL e un sistema ricevente di reporting che utilizza una connessione https e un modulo REST API per processare un payload in formato JSON



UNIVERSITÀ DEGLI STUDI DI TRIESTE

Un esempio pratico di integrazione

UN ESEMPIO PRATICO DI INTEGRAZIONE

Descrizione

Riprendiamo l'esempio precedente semplificandolo un po' ma andando nel dettaglio descrivendo la sua implementazione in Mirth Connect.

Supponiamo di avere un singolo channel in **Mirth** che riceva un flusso dati HL7 v.2 con messaggi di tipo ADT e ORM.

Il channel in questione filtrerà in base al tipo di messaggio ed eseguirà le seguenti azioni:

- Inoltro dei messaggi verso una destinazione remota utilizzando il protocollo http.
- Scrittura dei dati in un database locale di tipo SQL.

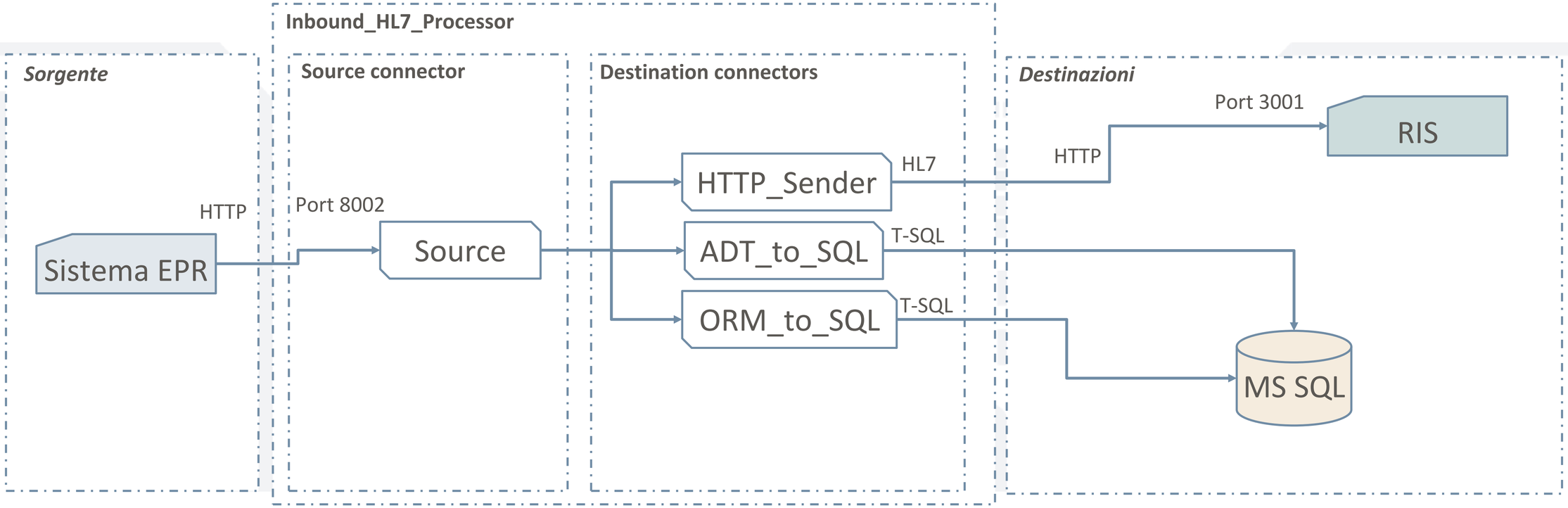
Tutti gli strumenti utilizzati per la realizzazione di questa interfaccia sono open source oppure già disponibili in ambiente Windows.

UN ESEMPIO PRATICO DI INTEGRAZIONE

Schema dell'interfaccia

Un ospedale utilizza un'instanza di Mirth Connect per ricevere un flusso dati HL7 da un sistema sorgente EPR. I messaggi sono inoltrati verso un sistema ricevente di radiologia (RIS) e il contenuto è successivamente archiviato in un database di tipo SQL.

In Mirth andremo ad implementare un channel con un *source connector* di tipo *http listener* e tre distinti *destination connectors*: un *http sender* per l'inoltro verso il sistema RIS e due destination di tipo *JavaScript Writer* per la scrittura dati nel database.

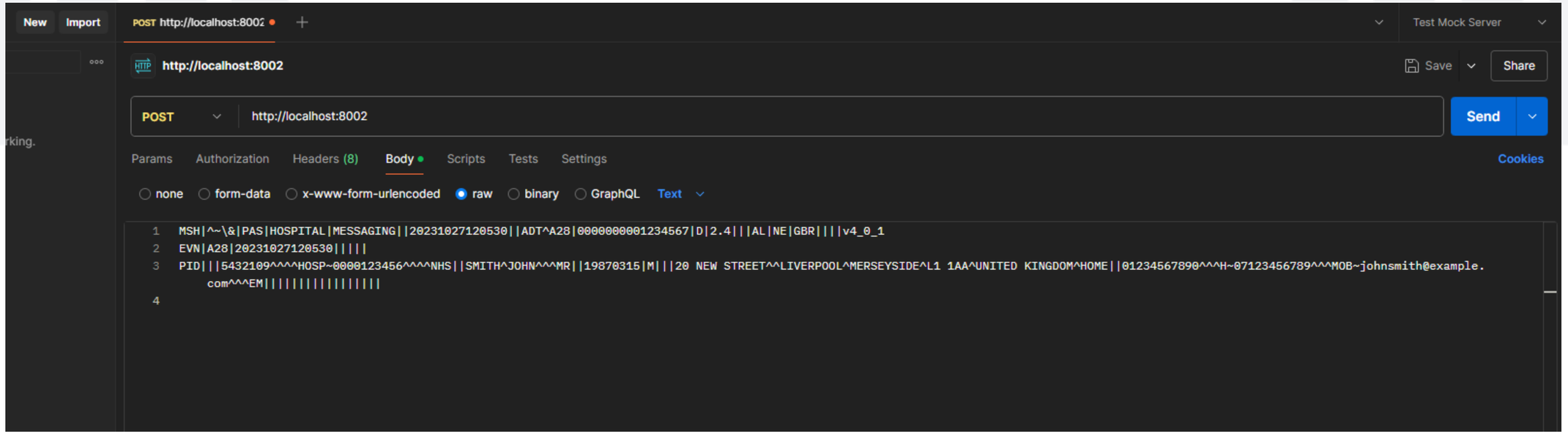


UN ESEMPIO PRATICO DI INTEGRAZIONE

Simulazione del sistema sorgente EPR

Per praticità possiamo usare **Postman** per inviare un messaggio HL7 localmente verso la porta **8002** utilizzando il protocollo HTTP. Postman è un tool scaricabile gratuitamente che permette di simulare il comportamento di un'interfaccia API (*application programming interface*) sia per ricevere che per inviare dati (https://www.postman.com).

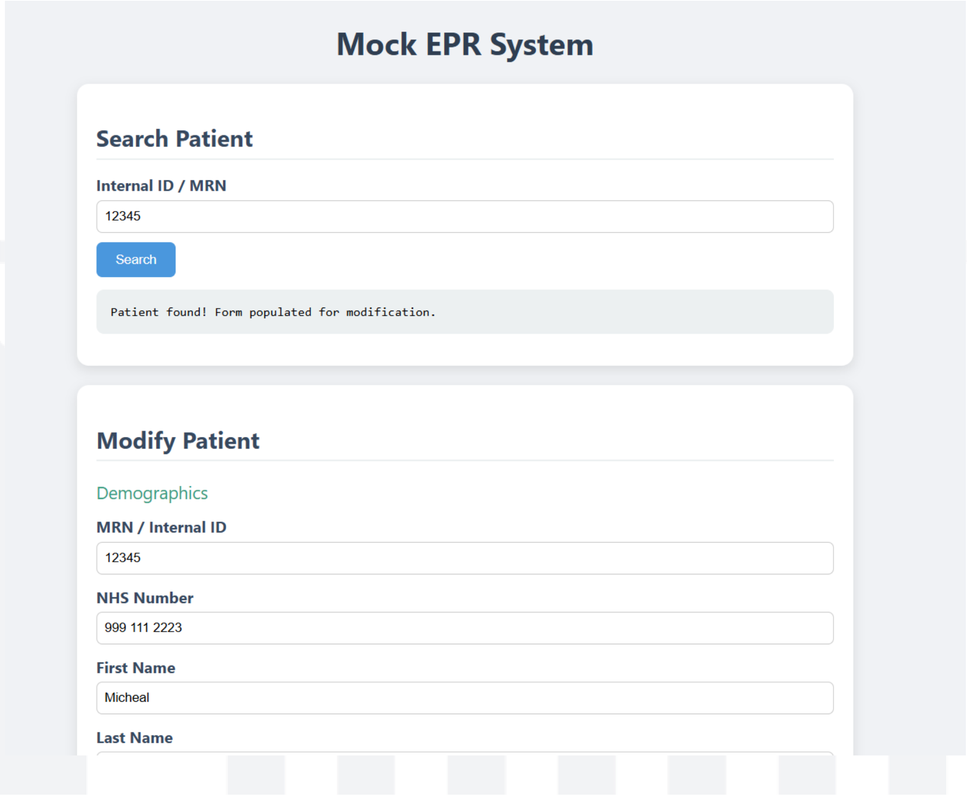
Il protocollo HTTP (*Hypertext Transfer Protocol*) è usato per la comunicazione su reti, principalmente per trasferire dati tra client e server sul web. Nelle richieste **POST**, i dati vengono inviati nel corpo della richiesta anziché nell'URL, permettendo di trasmettere informazioni complesse (es. form, files, messaggi HL7 etc.). Quando un client effettua una richiesta **POST** a un server su una porta specifica (solitamente la 80 per HTTP o 443 per HTTPS, in questo caso la porta 8002), il server riceve, elabora e risponde in base ai dati trasmessi.



UN ESEMPIO PRATICO DI INTEGRAZIONE

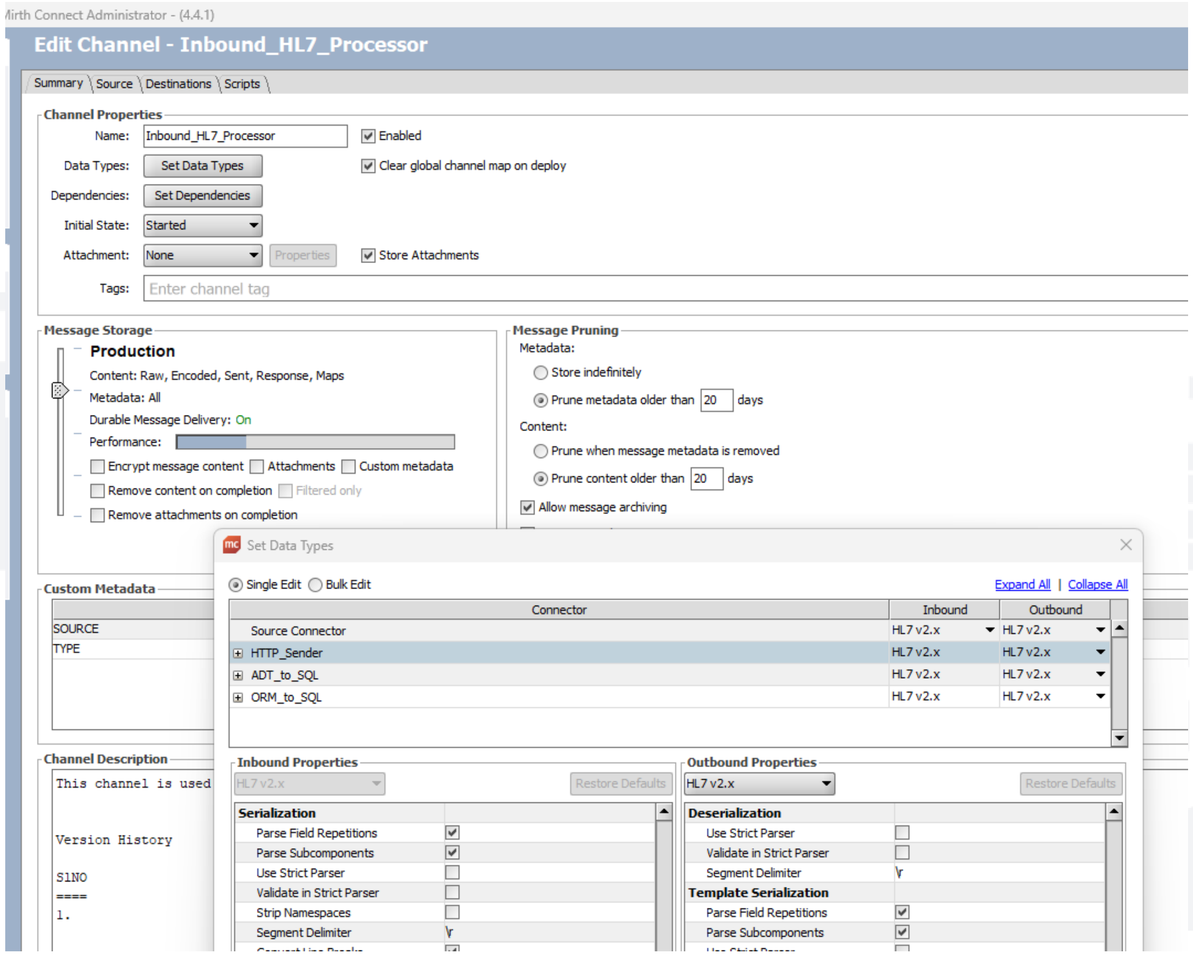
Simulazione del sistema sorgente EPR

Alternativamente possiamo anche utilizzare una semplice applicazione in *node.js* che simula la funzionalità di registrazione/modifica paziente ed invia i dati in formato HL7 via http.



UN ESEMPIO PRATICO DI INTEGRAZIONE

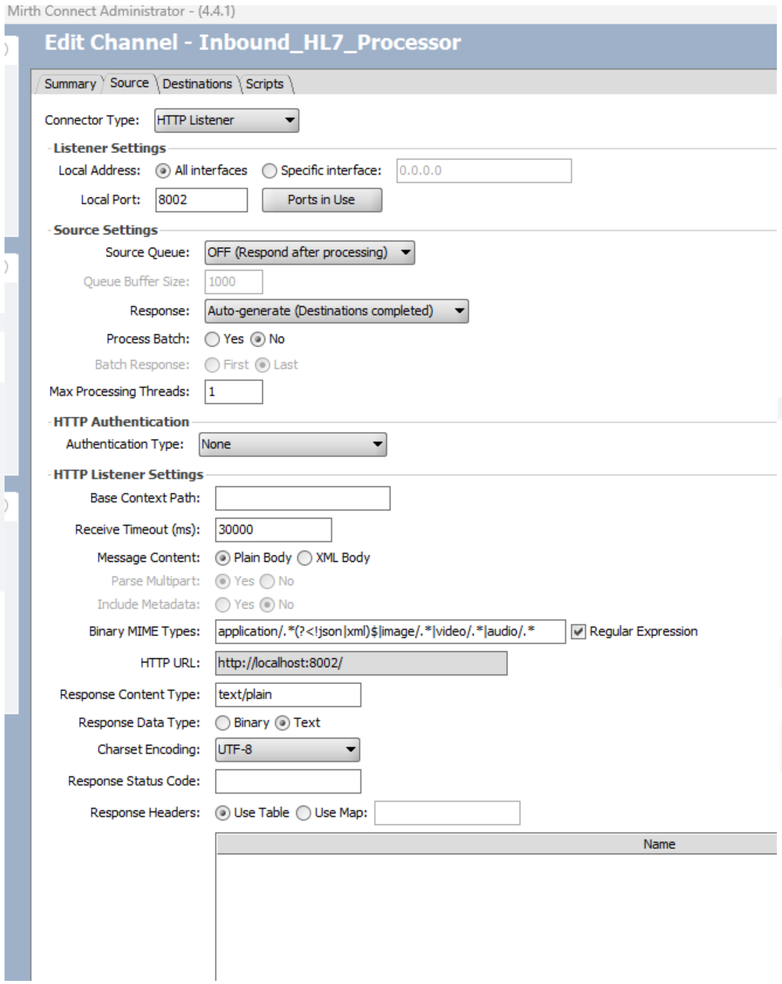
Summary



Nella *Summary* tab è possibile impostare settaggi generali come ad esempio il periodo di ritenzione dei messaggi nel database interno di Mirth, il *data type* di default dei source e *destination connectors*, lo stato iniziale del channel nonché' l'assegnazione del nome del channel stesso.

UN ESEMPIO PRATICO DI INTEGRAZIONE

Source connector



La configurazione del *source connector* prevede innanzitutto la selezione del *Connector Type*, in questo caso *HTTP Listener*.

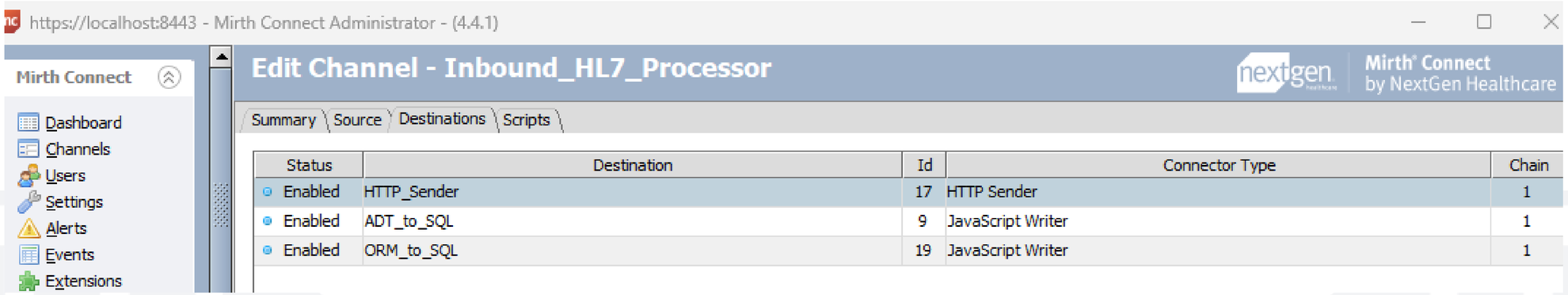
Successivamente si procede ad assegnare il valore per la porta in uso (**8002**) mentre tutti gli altri settaggi rimangono di default.

Il tipo di codifica utilizzata per i caratteri è *UTF-8*.

Non è prevista alcuna autenticazione.

UN ESEMPIO PRATICO DI INTEGRAZIONE

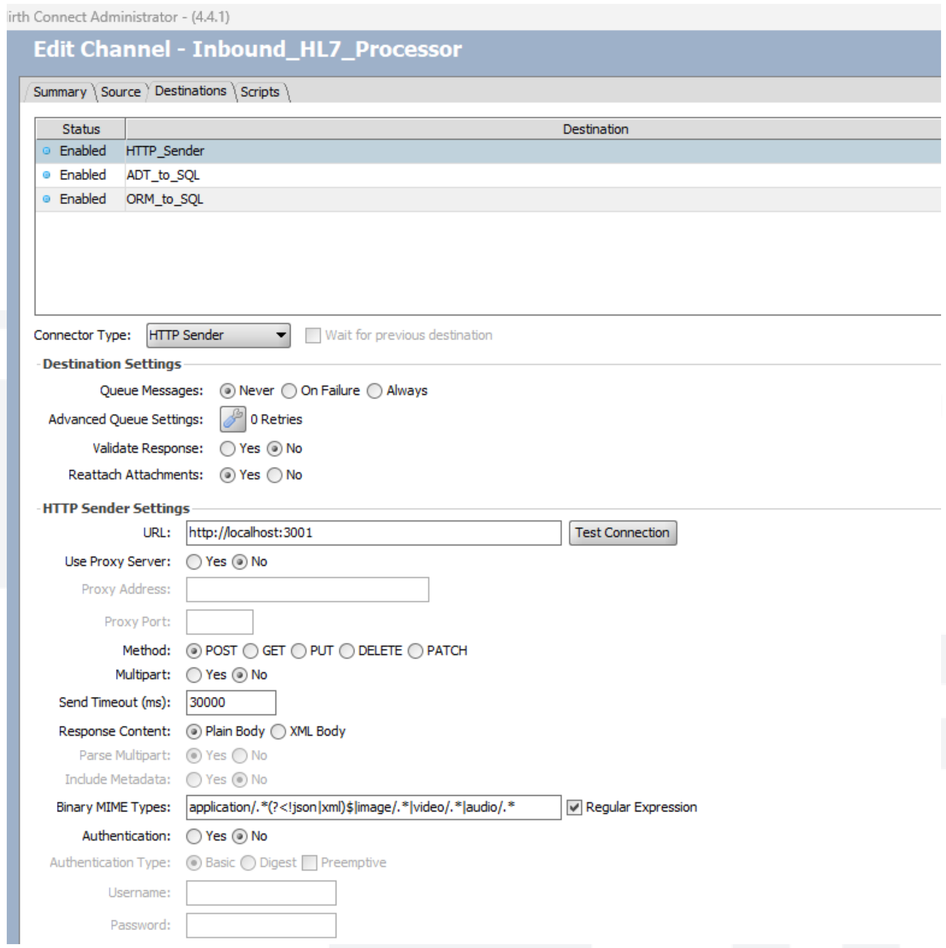
Overview sui connettori



Spostandoci sulla *Destinations* tab, possiamo quindi osservare i *destination connectors*. Come detto, sono tre. Si noti la tipologia dei connettori utilizzata, un connettore di tipo *HTTP Sender* e due connettori di tipo *JavaScript Writer*, che ora andremo a vedere nel dettaglio.

UN ESEMPIO PRATICO DI INTEGRAZIONE

HTTP Sender (1)



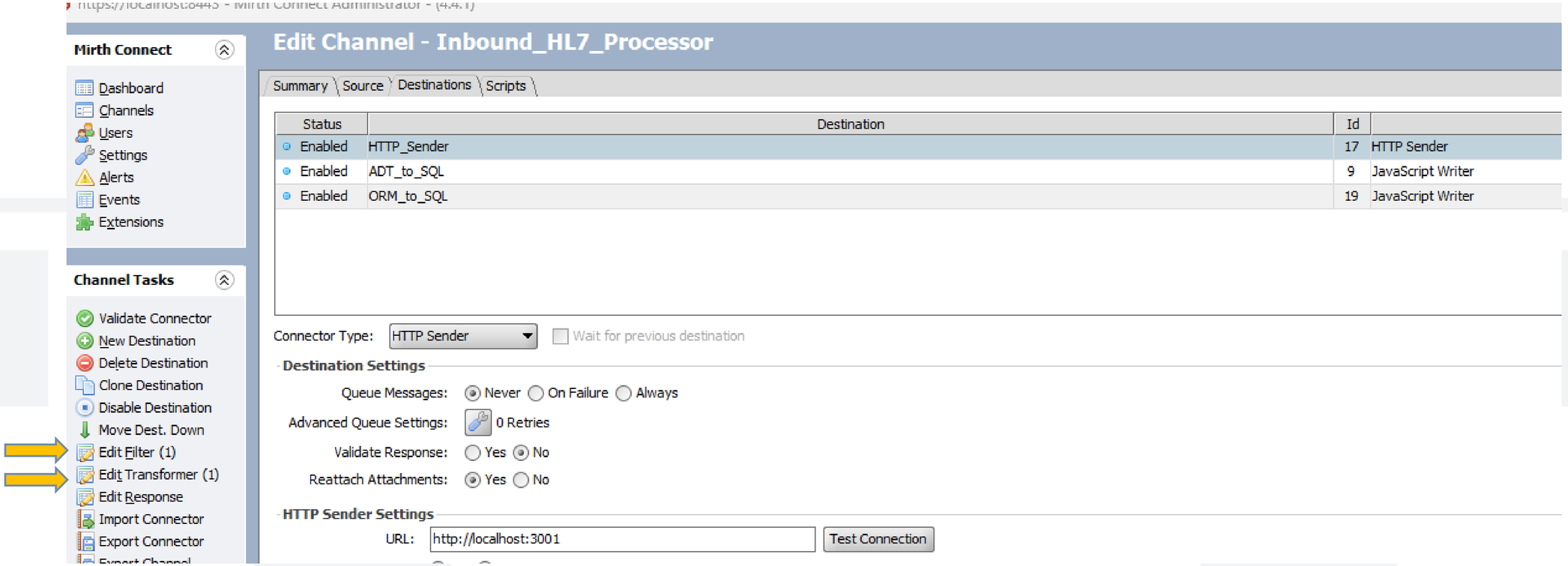
Il *destination connector* di tipo *HTTP sender* invia verso la porta **3001**.

Si noti come la destinazione è il *localhost*, semplicemente significa che per praticità inoltriamo i messaggi HL7 verso una destinazione sullo stesso server, in questo caso rappresentato dal mio portatile.

Come detto precedentemente, ipotizziamo dunque che il sistema ricevente sia un RIS dotato di un ricevitore HTTP.

UN ESEMPIO PRATICO DI INTEGRAZIONE

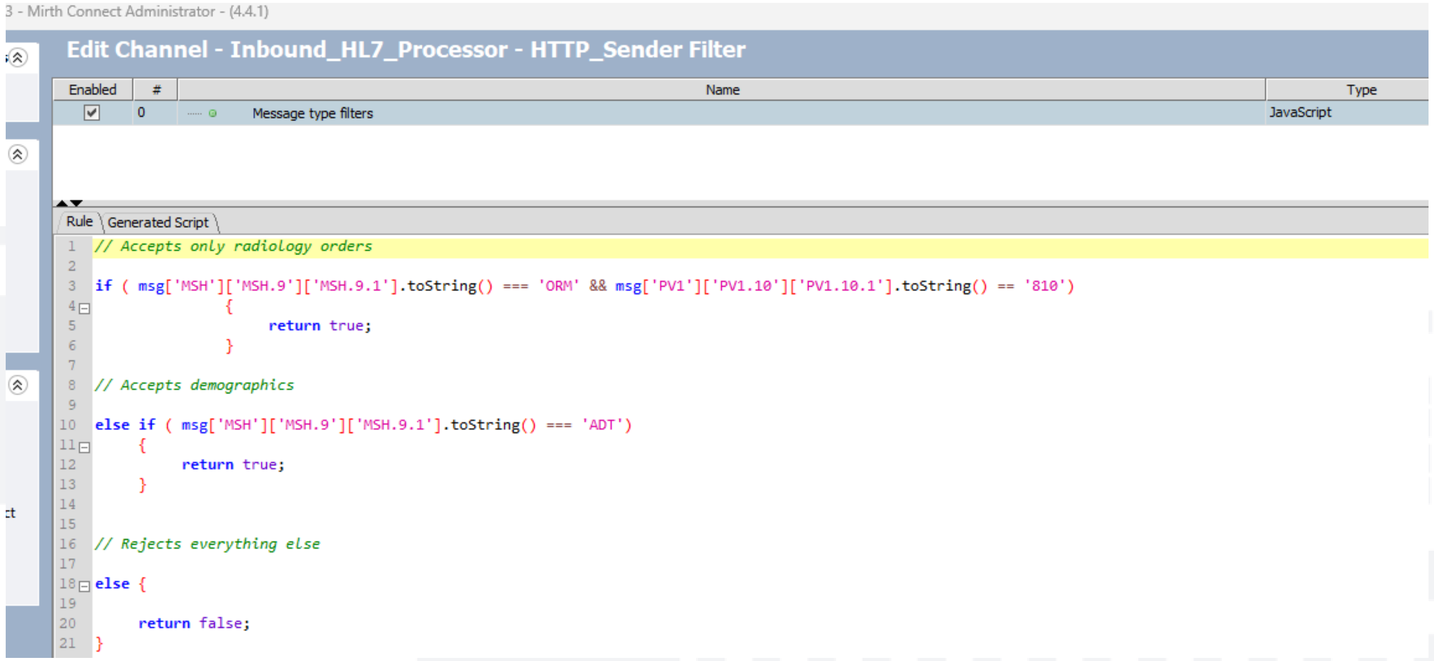
HTTP Sender (2)



Si noti la presenza di 1 filtro e di 1 transformer. Andiamo a vedere nel dettaglio cosa fanno.

UN ESEMPIO PRATICO DI INTEGRAZIONE

Il filtro dell'HTTP Sender



Il filtro è scritto direttamente in *JavaScript*, senza utilizzare i tools di Mirth.

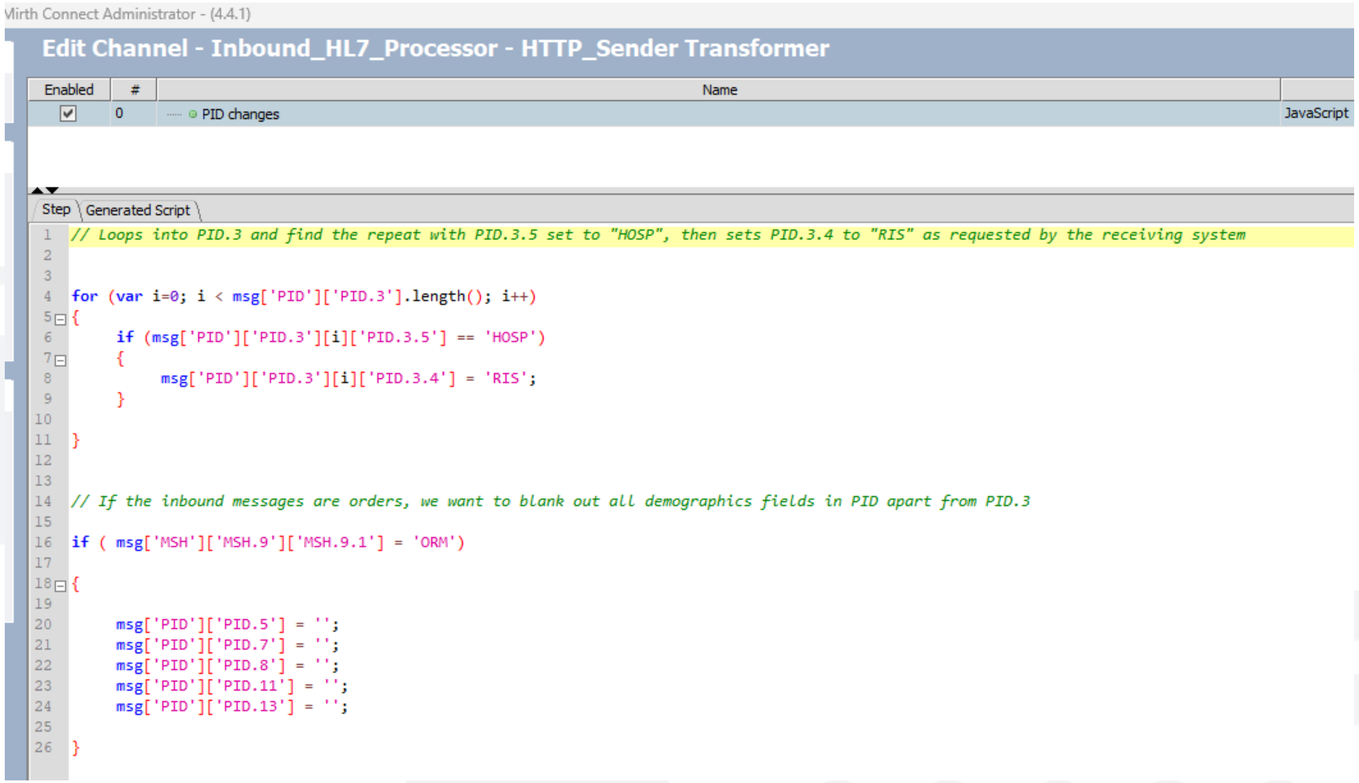
Vengono impostate due condizioni:

1) Messaggi di tipo **ORM** dove il codice in *PV1.10* è **810** (identificativo in UK per radiologia)
2) Messaggi di anagrafica di tipo **ADT**

Tutte le altre tipologie di messaggio non vengono processate.

UN ESEMPIO PRATICO DI INTEGRAZIONE

Il transformer dell'HTTP Sender



Il transformer viene utilizzato per cambiare la struttura del messaggio in uscita.

Si utilizza nuovamente *JavaScript* per inserire ad esempio un ulteriore codice identificativo per il paziente nel campo *PID.3.4* qualora il valore di *PID.3.5* sia **HOSP,** supponendo che le specifiche del sistema ricevente richiedano tale valore per processare il messaggio.

Inoltre, se il messaggio è un ordine, viene richiesto di non impostare nessun valore nei campi del segmento *PID* con la sola eccezione degli identificativi in *PID.3*

(vedremo un esempio end-to-end più avanti!)

UN ESEMPIO PRATICO DI INTEGRAZIONE

Simulazione del sistema ricevente RIS (1)

```powershell
# Create a listener for HTTP requests on localhost:3001
$listener = New-Object System.Net.HttpListener
$listener.Prefixes.Add("http://localhost:3001/")
$listener.Start()

Write-Host "Listening for requests on http://localhost:3001/..."

while ($listener.IsListening) {
    # Wait for an incoming request
    $context = $listener.GetContext()

    # Retrieve the request and response objects
    $request = $context.Request
    $response = $context.Response

    # Read the incoming request body
    $streamReader = New-Object System.IO.StreamReader($request.InputStream, $request.ContentEncoding)
    $payload = $streamReader.ReadToEnd()
    $streamReader.Close()

    # Format the HL7 message by replacing segment delimiters with newlines
    $formattedPayload = $payload -replace "`r", "`n"

    # Output the formatted payload to the PowerShell session
    Write-Host "Received payload:"
    Write-Host $formattedPayload

    # Prepare the response (JSON data)
    $jsonResponse = @{
        status  = "Success"
        message = "Received your request!"
    } | ConvertTo-Json

    # Set response headers
    $response.StatusCode = 200
    $response.ContentType = "application/json"

    # Write the response body
    $buffer = [System.Text.Encoding]::UTF8.GetBytes($jsonResponse)
    $response.OutputStream.Write($buffer, 0, $buffer.Length)

    # Close the response
    $response.Close()
}

# Stop the listener when finished
$listener.Stop()
```

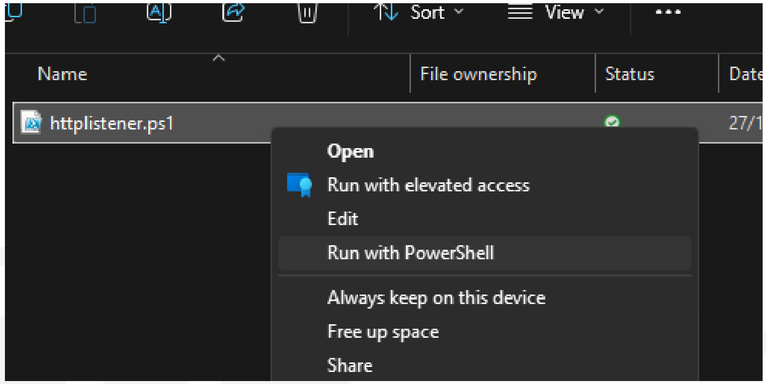
Utilizziamo un semplice script in PowerShell per simulare un ricevitore HTTP settato sulla porta **3001**.

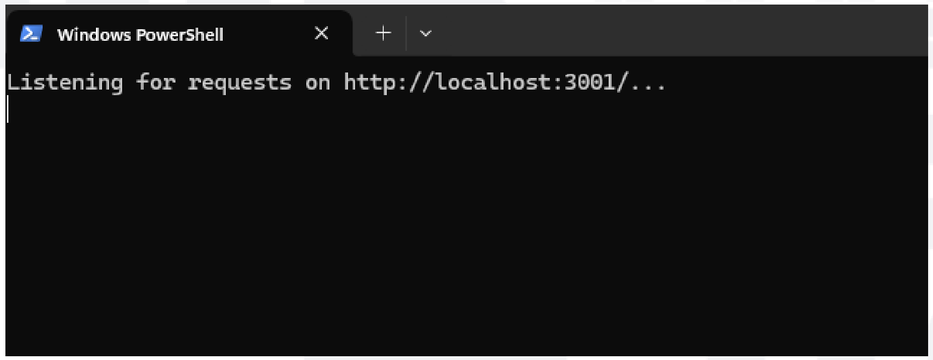
Utilizzando circa 40 linee di codice possiamo salvare un file di testo cambiando l'estensione del file in .**ps1** (estensione nativa di uno script PowerShell).

PowerShell è uno strumento disponibile in ambiente Windows e non necessita di installazione.

UN ESEMPIO PRATICO DI INTEGRAZIONE

Simulazione del sistema ricevente RIS (2)





Una volta creato il file, nell'esempio rinominato come **httplistener.ps1**, è sufficiente cliccare col tasto destro e scegliere dal menu l'opzione *Run with PowerShell*.

Si aprirà automaticamente una sessione PowerShell e il nostro dummy listener sarà adesso in grado di ricevere messaggi.

UN ESEMPIO PRATICO DI INTEGRAZIONE

Un cenno a Microsoft SQL Server

Le successive destinazioni andranno a scrivere in un database **Microsoft SQL**

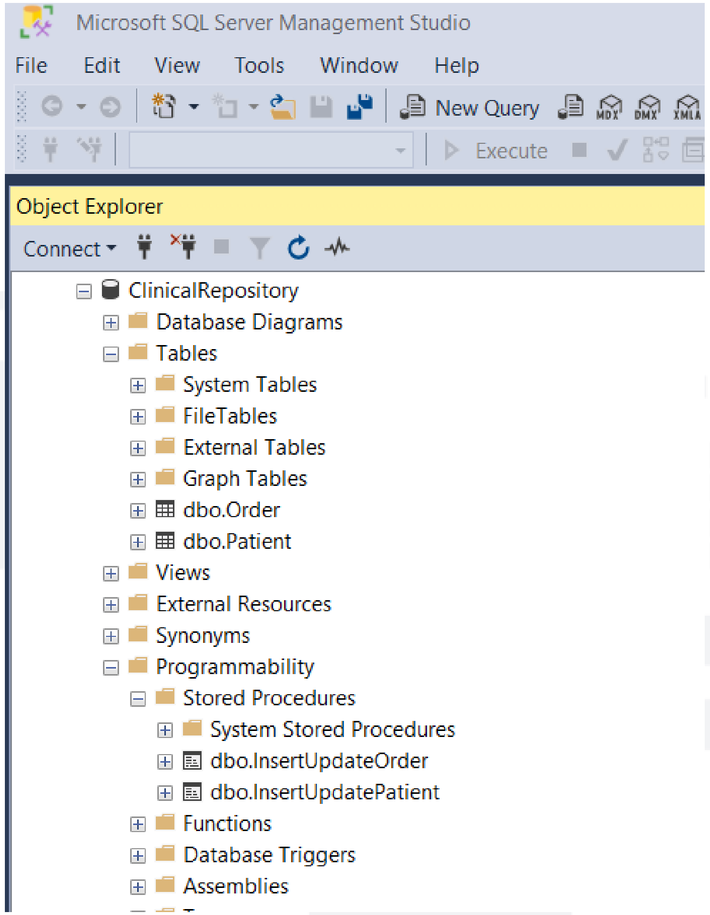
Microsoft SQL Server è la principale risorsa per database relazionali utilizzata su larga scala

Si può intendere un database relazionale come un insieme di tabelle legate tra loro utilizzando chiavi primarie e chiavi esterne contenenti dati e le istruzioni per la loro gestione raggruppate da uno schema. Il linguaggio utilizzato per la gestione di *Microsoft SQL Server* è **T-SQL** (*transact SQL*)

E’ possibile scaricare una versione free to use detta *SQL Express* che permette di installare un server **MS SQL** localmente.

UN ESEMPIO PRATICO DI INTEGRAZIONE

Struttura di un database SQL



Si può accedere un database MS SQL utilizzando il client *Microsoft SQL Server Management Studio*.

Nell'esempio che andremo a utilizzare il nostro database contiene due tabelle, una contenente un elenco di pazienti (*dbo.Patient*) e un'altra contenente un elenco di ordini (*dbo.Order*).

Due *stored procedures* contengono invece le istruzioni in T-SQL che utilizzeremo per inserire dati nelle due tabelle attraverso il sistema di integrazione Mirth Connect.

UN ESEMPIO PRATICO DI INTEGRAZIONE

Struttura della tabella dbo.Patient

```
CREATE TABLE [dbo].[Patient](
    [PatientInternalId] [uniqueidentifier] NOT NULL,
    [HospitalNumber] [varchar](100) NOT NULL,
    [NHSNumber] [varchar](50) NULL,
    [Title] [varchar](50) NULL,
    [FirstName] [varchar](100) NULL,
    [LastName] [varchar](100) NULL,
    [DateOfBirth] [smalldatetime] NULL,
    [DateOfDeath] [smalldatetime] NULL,
    [Gender] [int] NOT NULL,
    [Street] [varchar](255) NULL,
    [City] [varchar](255) NULL,
    [Province] [varchar](255) NULL,
    [Country] [varchar](255) NULL,
    [PostCode] [varchar](20) NULL,
    [MobileNumber] [varchar](100) NULL,
    [SecondaryNumber] [varchar](100) NULL,
    [Email] [varchar](100) NULL,
    [CreatedOn] [datetime] NOT NULL,
    [UpdatedOn] [datetime] NOT NULL,
PRIMARY KEY CLUSTERED
(
    [PatientInternalId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = ON, ALLOW_ROW_LOCKS = ON,
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[Patient] ADD  DEFAULT (getdate()) FOR [CreatedOn]
GO

ALTER TABLE [dbo].[Patient] ADD  DEFAULT (getdate()) FOR [UpdatedOn]
GO
```

Nell’esempio a fianco osserviamo il codice T-SQL utilizzato per creare la tabella *dbo.Patient* con un elenco dei campi, ognuno costituente una colonna della tabella, e la tipologia di dato contenuto.

UN ESEMPIO PRATICO DI INTEGRAZIONE

Struttura della tabella dbo.Order

```
CREATE TABLE [dbo].[Order](
    [OrderInternalId] [uniqueidentifier] NOT NULL,
    [HospitalNumber] [varchar](100) NOT NULL,
    [NHSNumber] [varchar](50) NULL,
    [RequestedOn] [smalldatetime] NULL,
    [OrderPlacerId] [varchar](50) NULL,
    [OrderFillerId] [varchar](50) NULL,
    [ProcedureDate] [smalldatetime] NULL,
    [Modality] [varchar](255) NULL,
    [RequestedById] [varchar](50) NULL,
    [Diagnosis] [varchar](255) NULL,
    [VisitId] [varchar](50) NULL,
    [SpecialtyId] [varchar](3) NULL,
    [PatientClass] [varchar](1) NULL,
    [CreatedOn] [datetime] NOT NULL,
    [UpdatedOn] [datetime] NOT NULL,
PRIMARY KEY CLUSTERED
(
    [OrderInternalId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = ON, ALLOW_ROW_LOCKS = ON, ALLOW_PA
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[Order] ADD  DEFAULT (getdate()) FOR [CreatedOn]
GO

ALTER TABLE [dbo].[Order] ADD  DEFAULT (getdate()) FOR [UpdatedOn]
GO
```

Similmente, nell'esempio a fianco osserviamo il codice T-SQL utilizzato per creare la tabella *dbo.Order* con un elenco dei campi, ognuno costituente una colonna della tabella, e la tipologia di dato contenuto.

UN ESEMPIO PRATICO DI INTEGRAZIONE

Struttura della stored procedure dbo.InsertUpdatePatient

```
ALTER PROCEDURE [dbo].[InsertUpdatePatient]
    @HospitalNumber VARCHAR(100),
    @NHSNumber VARCHAR(50) = NULL,
    @Title VARCHAR(50) = NULL,
    @FirstName VARCHAR(100) = NULL,
    @LastName VARCHAR(100) = NULL,
    @DateOfBirth SMALLDATETIME = NULL,
    @DateOfDeath SMALLDATETIME = NULL,
    @Gender INT,
    @Street VARCHAR(255) = NULL,
    @City VARCHAR(255) = NULL,
    @Province VARCHAR(255) = NULL,
    @Country VARCHAR(255) = NULL,
    @PostCode VARCHAR(20) = NULL,
    @MobileNumber VARCHAR(100) = NULL,
    @SecondaryNumber VARCHAR(100) = NULL,
    @Email VARCHAR(100) = NULL
AS
BEGIN
    SET NOCOUNT ON;

    DECLARE @CurrentDateTime DATETIME = GETDATE();
    DECLARE @NewPatientInternalId UNIQUEIDENTIFIER;
```

```
    -- Check if a record exists based on HospitalNumber
    IF EXISTS (SELECT 1 FROM dbo.Patient WHERE HospitalNumber = @HospitalNumber)
    BEGIN
        -- Update the existing record
        UPDATE dbo.Patient
        SET
            NHSNumber = @NHSNumber,
            Title = @Title,
            FirstName = @FirstName,
            LastName = @LastName,
            DateOfBirth = @DateOfBirth,
            DateOfDeath = @DateOfDeath,
            Gender = @Gender,
            Street = @Street,
            City = @City,
            Province = @Province,
            Country = @Country,
            PostCode = @PostCode,
            MobileNumber = @MobileNumber,
            SecondaryNumber = @SecondaryNumber,
            Email = @Email,
            UpdatedOn = @CurrentDateTime
        WHERE HospitalNumber = @HospitalNumber;
    END
```

```
    ELSE
    BEGIN
        -- Generate a new GUID for PatientInternalId
        SET @NewPatientInternalId = NEWID();

        -- Insert a new record
        INSERT INTO dbo.Patient (
            PatientInternalId,
            HospitalNumber,
            NHSNumber,
            Title,
            FirstName,
            LastName,
            DateOfBirth,
            DateOfDeath,
            Gender,
            Street,
            City,
            Province,
            Country,
            PostCode,
            MobileNumber,
            SecondaryNumber,
            Email,
            CreatedOn,
            UpdatedOn
        )
        VALUES (
            @NewPatientInternalId,
            @HospitalNumber,
            @NHSNumber,
            @Title,
            @FirstName,
            @LastName,
            @DateOfBirth,
            @DateOfDeath,
            @Gender,
            @Street,
            @City,
            @Province,
```

La **stored procedure** *dbo.InsertUpdatePatient* contiene le istruzioni in T-SQL per inserire un nuovo record nella tabella per ognuno dei campi previsti. Se il record è stato precedentemente ricevuto, si procede ad un update altrimenti si fa un nuovo inserimento.

UN ESEMPIO PRATICO DI INTEGRAZIONE

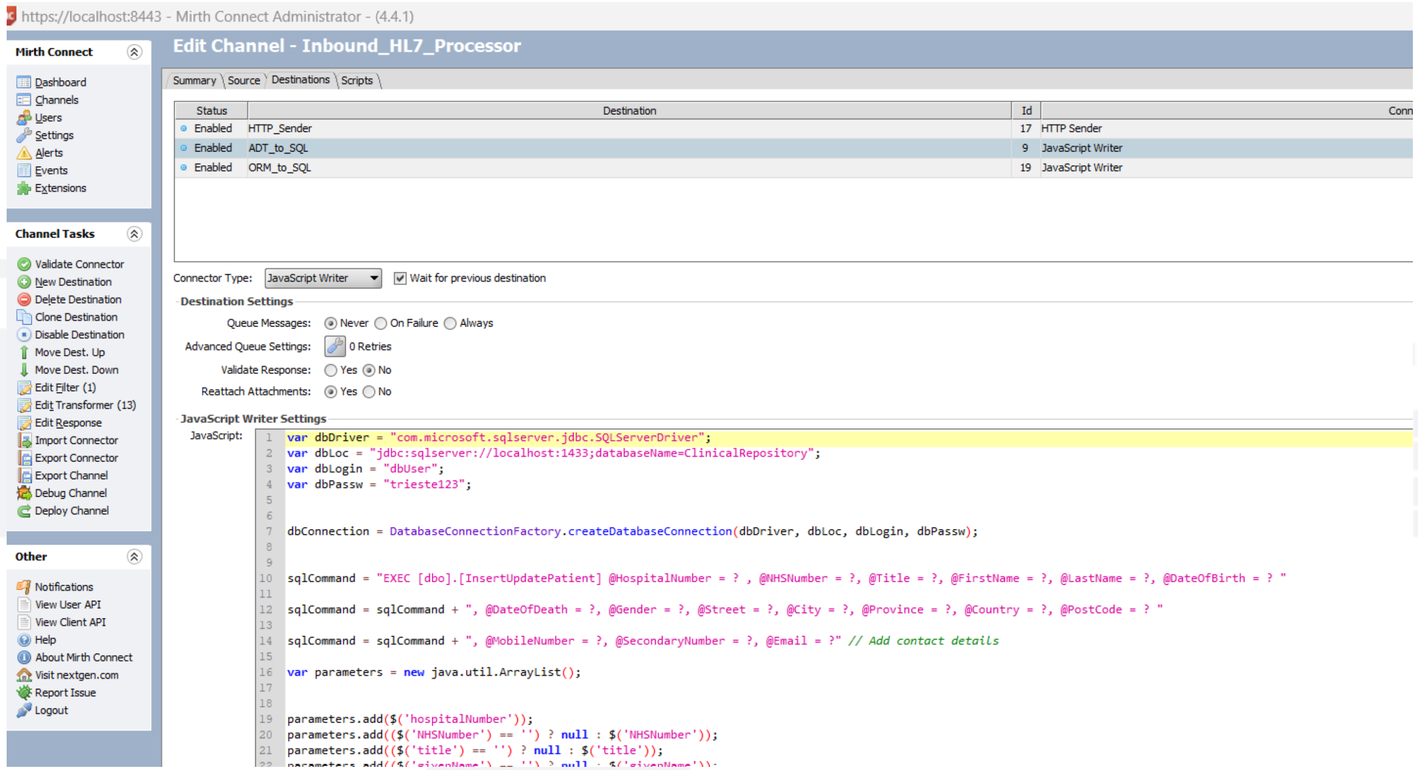
Struttura della stored procedure dbo.InsertUpdateOrder

```
ALTER PROCEDURE [dbo].[InsertUpdateOrder]
    @HospitalNumber VARCHAR(100),
    @NHSNumber VARCHAR(50) = NULL,
    @RequestedOn SMALLDATETIME = NULL,
    @OrderPlacerId VARCHAR(50) = NULL,
    @OrderFillerId VARCHAR(50) = NULL,
    @ProcedureDate SMALLDATETIME = NULL,
    @Modality VARCHAR(255) = NULL,
    @RequestedById VARCHAR(50) = NULL,
    @Diagnosis VARCHAR(255) = NULL,
    @VisitId VARCHAR(50) = NULL,
    @SpecialtyId VARCHAR(3) = NULL,
    @PatientClass VARCHAR(1) = NULL
AS
BEGIN
    SET NOCOUNT ON;

    DECLARE @CurrentDateTime DATETIME = GETDATE();
    DECLARE @NewOrderInternalId UNIQUEIDENTIFIER;

    -- Check if a record exists based on HospitalNumber
    IF EXISTS (SELECT 1 FROM dbo.[Order] WHERE OrderPlacerId = @OrderPlacerId)
    BEGIN
        -- Update the existing record
        UPDATE dbo.[Order]
        SET
            HospitalNumber = @HospitalNumber,
            NHSNumber = @NHSNumber,
            RequestedOn = @RequestedOn,
            OrderFillerId = @OrderFillerId,
            ProcedureDate = @ProcedureDate,
            Modality = @Modality,
            RequestedById = @RequestedById,
            Diagnosis = @Diagnosis,
            VisitId = @VisitId,
            SpecialtyId = @SpecialtyId,
            PatientClass = @PatientClass,
            UpdatedOn = @CurrentDateTime

        WHERE OrderPlacerId = @OrderPlacerId;
    END
ELSE
BEGIN
    -- Generate a new GUID for OrderInternalId
    SET @NewOrderInternalId = NEWID();

    -- Insert a new record
    INSERT INTO dbo.[Order] (
        OrderInternalId,
        HospitalNumber,
        NHSNumber,
        RequestedOn,
        OrderPlacerId,
        OrderFillerId,
        ProcedureDate,
        Modality,
        RequestedById,
        Diagnosis,
        VisitId,
        SpecialtyId,
        PatientClass,
        CreatedOn,
        UpdatedOn
    )
    VALUES (
        @NewOrderInternalId,
        @HospitalNumber,
        @NHSNumber,
        @RequestedOn,
        @OrderPlacerId,
        @OrderFillerId,
        @ProcedureDate,
        @Modality,
```

La **stored procedure** *dbo.InsertUpdateOrder* contiene le istruzioni in T-SQL per inserire un nuovo record nella tabella per ognuno dei campi previsti. Se il record è stato precedentemente ricevuto, si procede ad un update altrimenti si fa un nuovo inserimento.

UN ESEMPIO PRATICO DI INTEGRAZIONE

Il destination connector ADT_to_SQL



Il *destination connector* **ADT_to_SQL** contiene le istruzioni in JavaScript per eseguire la **stored procedure** *dbo.InsertUpdatePatient*.

Mirth permette infatti di utilizzare drivers JDBC per inizializzare una connessione ad un database MS SQL ed eseguire un comando T-SQL

In questo caso andremo ad eseguire *EXEC dbo.InsertUpdatePatient* dove i parametri sono ottenuti leggendo i dati dai campi del messaggio HL7 ricevuto.

UN ESEMPIO PRATICO DI INTEGRAZIONE

Il destination connector ADT_to_SQL

Edit Channel - Inbound_HL7_Processor - ADT_to_SQL Transformer

Enabled	#	Name
☑	0	patient ID's
☑	1	gender
☑	2	familyName
☑	3	givenName
☑	4	title
☑	5	dateOfBirth
☑	6	dateOfDeath
☑	7	street
☑	8	city
☑	9	province
☑	10	country
☑	11	postCode
☑	12	contactDetails

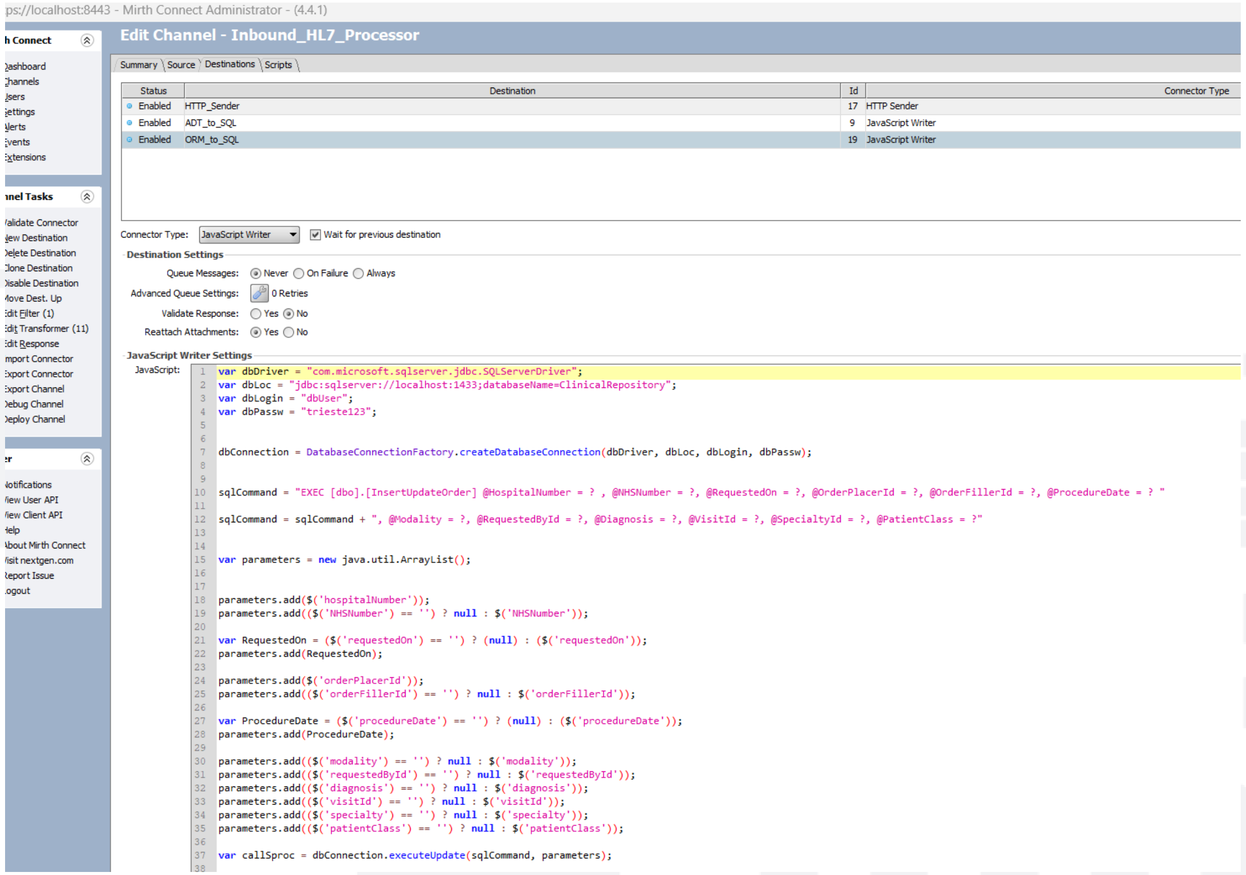
Step | Generated Script

```
1  var gender;
2  var genderCode = msg['PID']['PID.8']['PID.8.1'].toString();
3
4  if (String(genderCode) === "M") {
5      gender = 1;
6  } else if (String(genderCode) === "F") {
7      gender = 2;
8  } else if (String(genderCode) === "NS") {
9      gender = 9;
10 } else if (String(genderCode) === "U") {
11     gender = 0;
12 } else {
13     gender = -1;
14 }
15
16
17 channelMap.put('gender', validate(gender, '', new Array()));
```

Il *destination connector* **ADT_to_SQL** è caratterizzato da un filtro che accetta solo messaggi di tipo ADT e da una serie di transformers che sono appunto le istruzioni per leggere i dati dal messaggio HL7.

UN ESEMPIO PRATICO DI INTEGRAZIONE

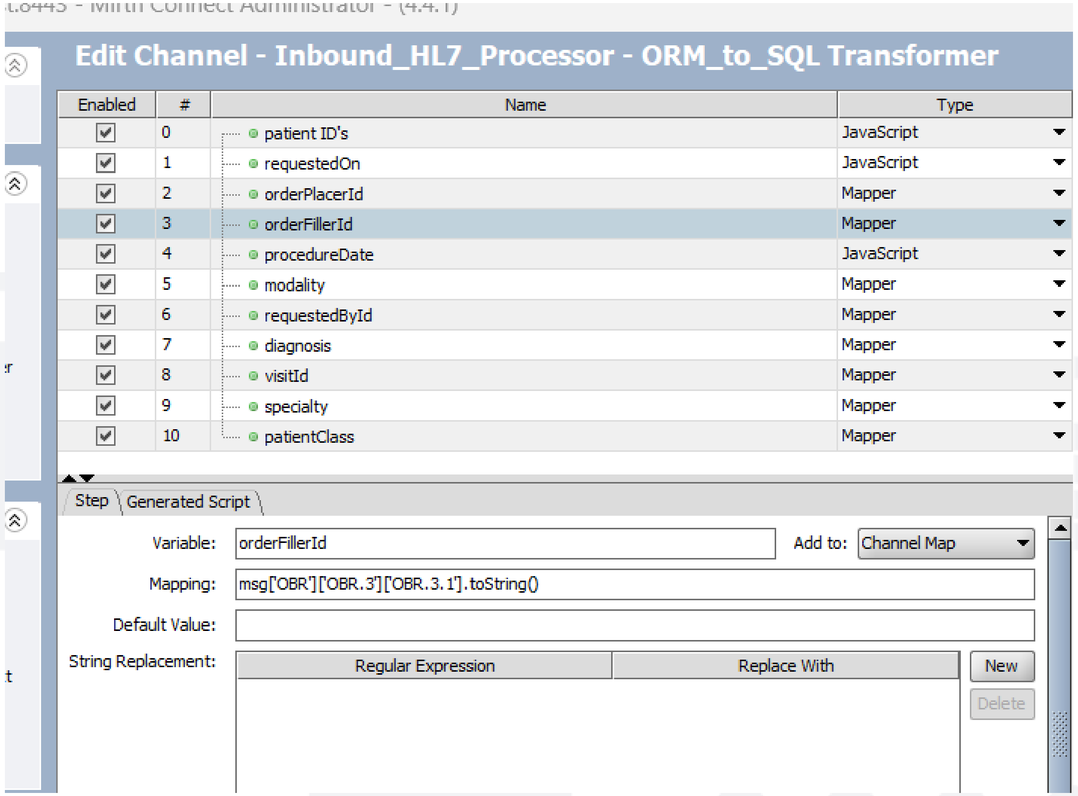
Il destination connector ORM_to_SQL



Similmente il *destination connector* **ORM_to_SQL** è caraterizzato da un filtro che accetta solo messaggi di tipo ORM e da una serie di transformers che sono appunto le istruzioni per leggere i dati dal messaggio HL7 (**ORM^O01**).

UN ESEMPIO PRATICO DI INTEGRAZIONE

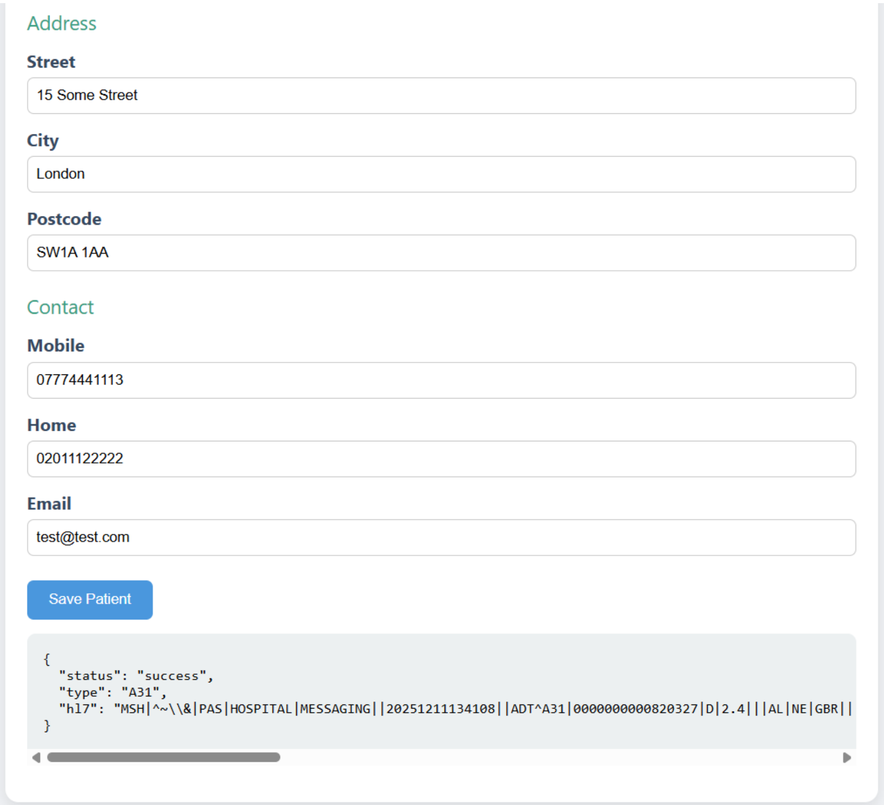
Il destination connector ORM_to_SQL



Nell'esempio di transformer qui a fianco andiamo a leggere l'*order filler ID* utilizzando un transformer di tipo *Mapper* anziché scrivere del codice in *JavaScript*.

DIMOSTRAZIONE END-TO-END

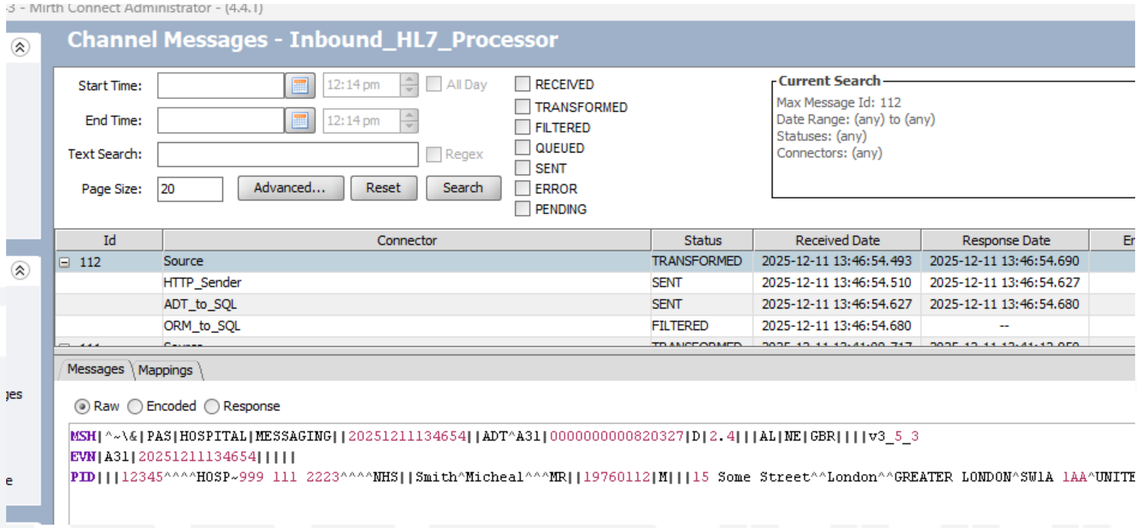
Invio del messaggio HL7

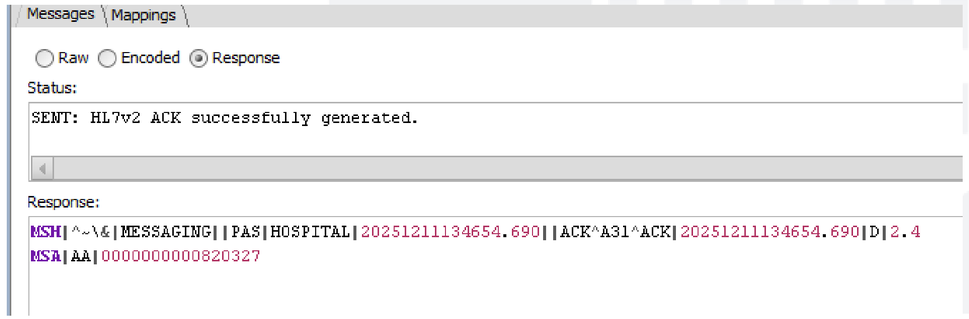


Simuliamo l'invio di un messaggio **ADT** utilizzando un webform. Dopo aver inviato il messaggio possiamo osservare l'acknowledgment nel webform in formato JSON

DIMOSTRAZIONE END-TO-END

Ricezione del messaggio HL7



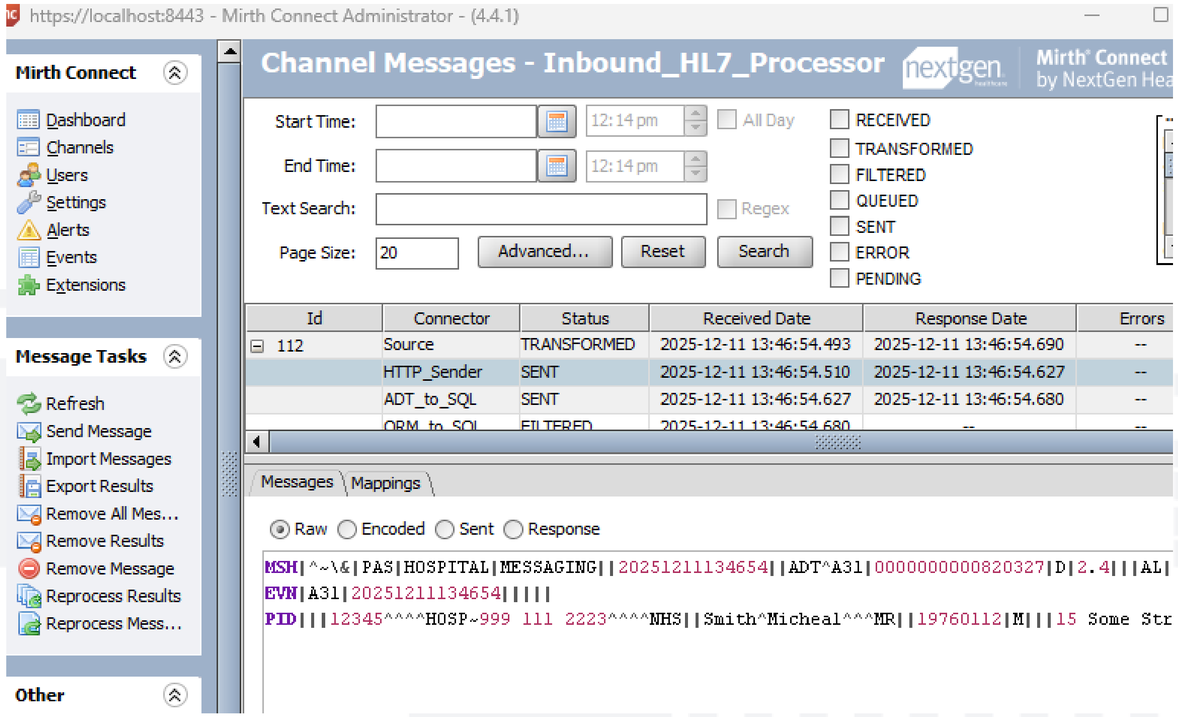


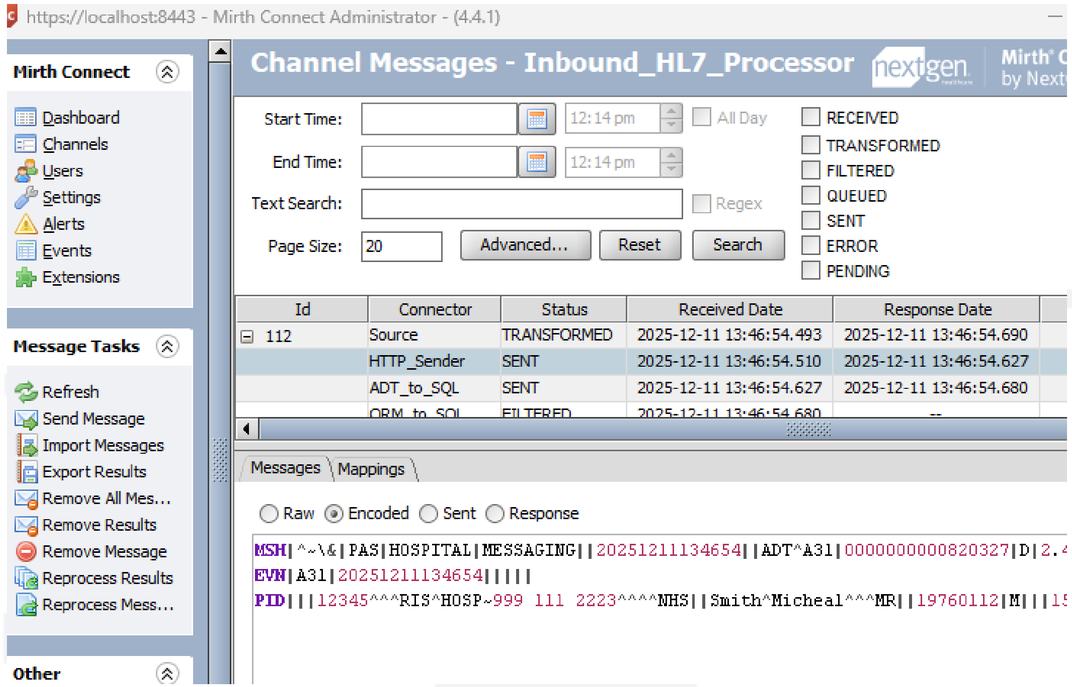
Nel dashboard possiamo osservare come il messaggio sia stato correttamente filtrato dalla destinazione ORM_to_SQL.

Inoltre, come già visto in **Postman**, Mirth ha generato correttamente la «ricevuta», ovvero il messaggio di **ACK** inviato al sistema sorgente, con codice **AA** (*application accept*) che permette di chiudere la transazione a monte. In caso di errore, un **NACK** con codice **AR** (*application reject*) può ad esempio essere utilizzato dal sistema sorgente come istruzione per inviare il messaggio nuovamente.

DIMOSTRAZIONE END-TO-END

Inoltro del messaggio HL7 verso la destinazione http

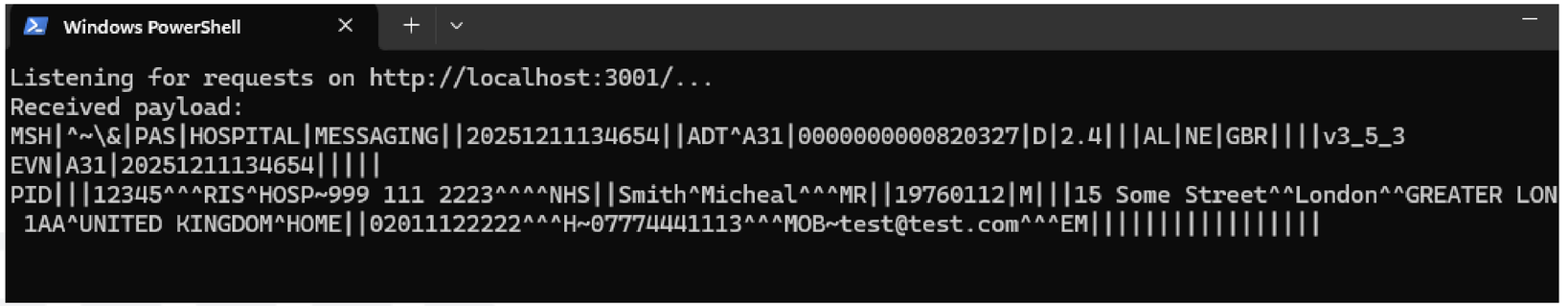


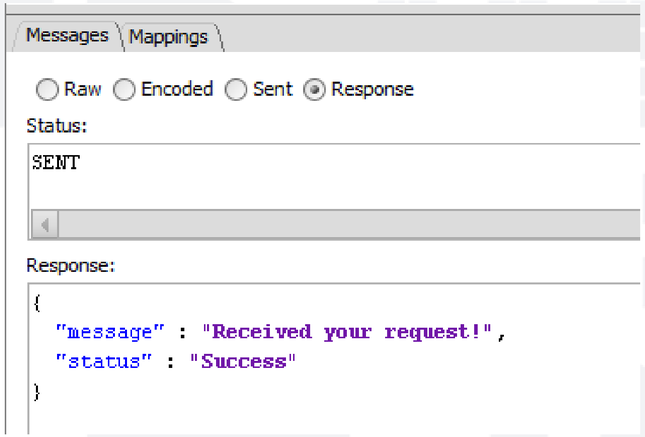


Il messaggio viene trasformato prima di essere inviato verso la destinazione http con l'aggiunta di un valore *hard coded* in **PID.3.4** dell'*hospital number* che sarà utilizzato dal sistema ricevente per inserire il record nel sistema.

DIMOSTRAZIONE END-TO-END

Ricezione del messaggio nella destinazione http



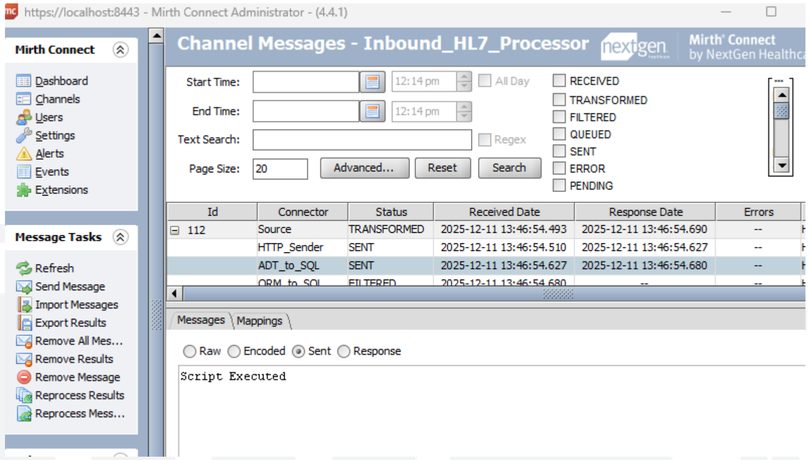


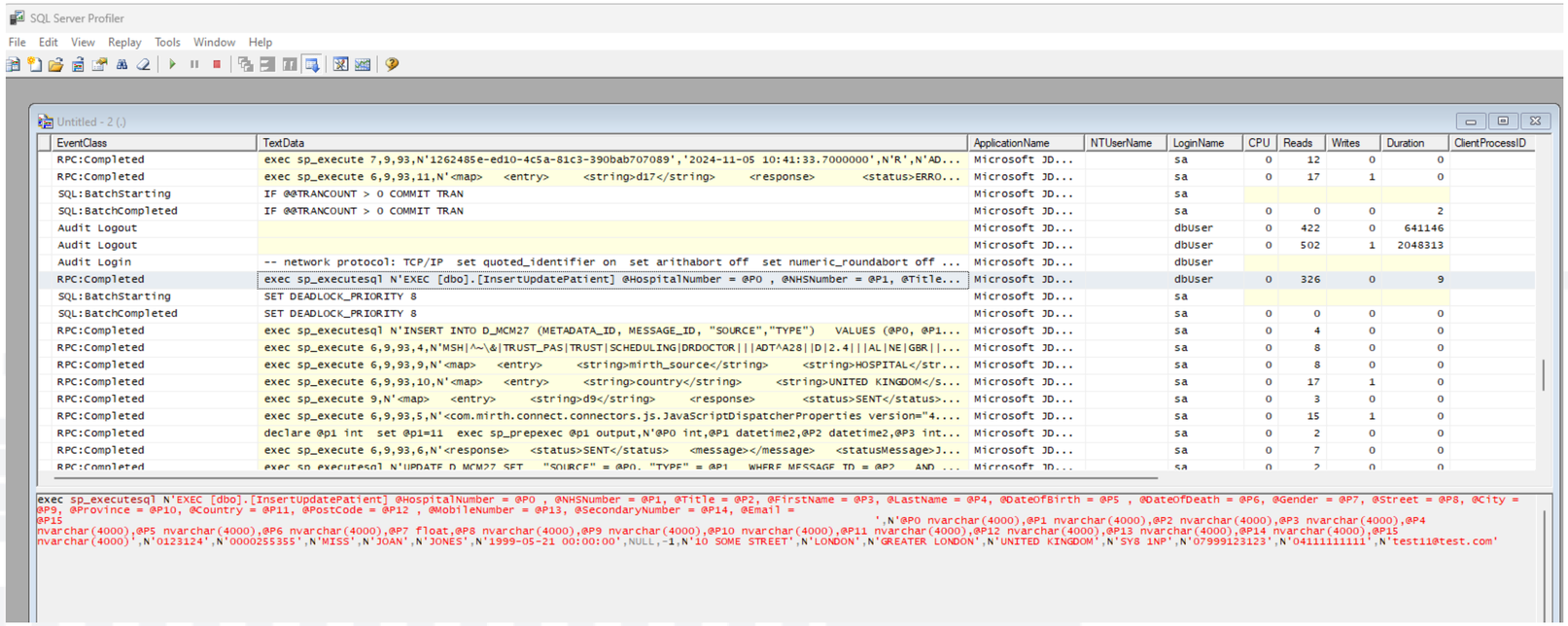
Il messaggio viene ricevuto e visualizzato in powershell. Si noti il valore aggiuntivo nel campo **PID.3.4**.

Il sistema ricevente invia inoltre una risposta di tipo JSON che viene ricevuta e visualizzata in Mirth.

DIMOSTRAZIONE END-TO-END

Scrittura del record nel database SQL

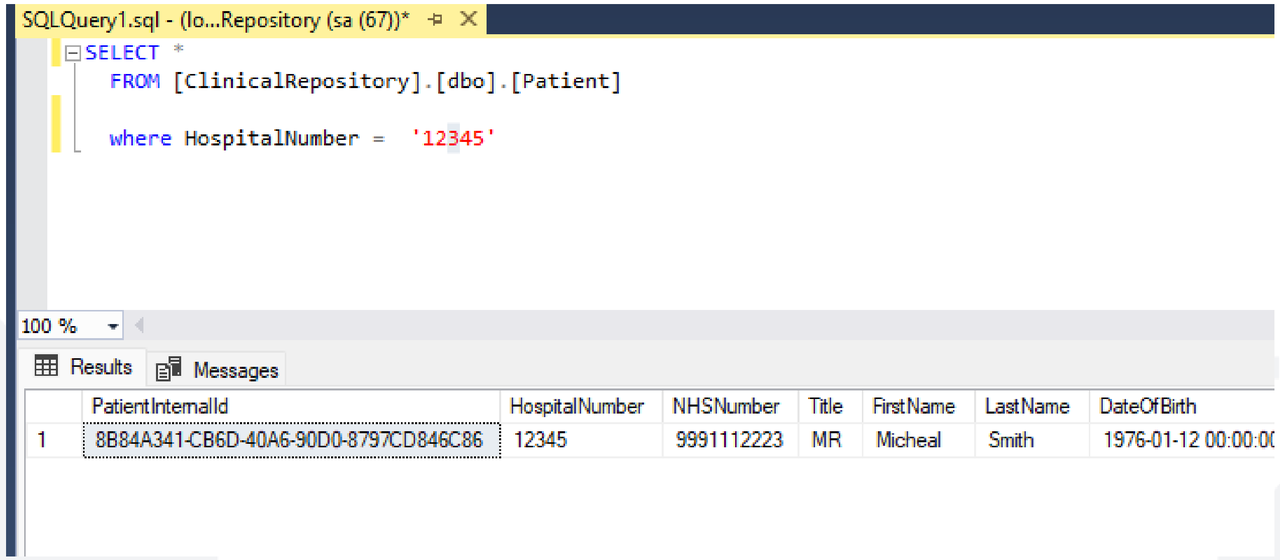




Mirth conferma che lo script *JavaScript* è stato eseguito correttamente senza errori. Possiamo utilizzare il tool *SQL Server Profiler* per tracciare l'attività nel database e osservare la ricezione e l'esecuzione del comando **T-SQL** che andrà ad eseguire la stored procedure *InsertUpdatePatient*, con i parametri ricevuti da Mirth.

DIMOSTRAZIONE END-TO-END

Lettura del record nel database SQL



Utilizzando *SQL Server Management Studio* possiamo quindi leggere il record appena inserito nella tabella *dbo.Patient* utilizzando una semplice query SQL.