HL7 FHIR Data Consolidation Tool

What is HL7 FHIR?

HL7 FHIR or FHIR (Fast Healthcare Interoperability Resources pronounced as "Fire") is a part of the Medical Healthcare information ecosystem that is a result of many years of work done by volunteers and healthcare companies, members of the HL7.org, the custodian body for all HL7 standards. They have put in considerable efforts on standardization of how information flows between different healthcare systems.

Different Version of HL7

FHIR is HL7 standard in healthcare that is the latest healthcare information technology standard that has all the best features of the previous HL7 standardization efforts such as HL7 2.x and HL7 3.0 and HL7 CDA (Clinical Document Architecture) and yet its use is more practical and developer friendly. The HL7 version 3.0 did not catch up as wildly as HL7 2.x did, despite that, it could be directly rendered in the browsers and had many other useful features. After the HL7.org did surveys of the organizations, they discovered that it was too expensive and complicated to implement these standards and therefore the HL7 3.0 was left out. It led the HL7.org back to the drawing board, and the executives, the technology decision maker, software architects and the developers worked together to come up with a standard that could deliver higher value, and FHIR standard came into existence in 2014. Since 2014, FHIR has been gaining popularity in the healthcare industry. Technosoft, one of the very few offshore technology firms specializing in healthcare integration services, was natural drawn to FHIR development and this article is written to provide details about one of such FHIR tool developed by Technosoft.

www.techno-soft.com | info@techno-soft.com | (203) 676-8299

Technosoft is creating the HL7 FHIR Data Driven Tool

The primary requirement of the Data Consolidation FHIR tool was to consolidate patient data from different sources and present it through a single FHIR RESTful API. The software tool was designed initially to parse CCD, CSV files, HL7 2.x files or FHIR records of various patients and then save it in a database where calculations and display of specific holistic information take place in an aggregated form. A user can make queries from the a database with an easy to use interface such as how many patients in a medical facility had been suffering from flu in a season? Or, how many patients were found with cancer in a specific year?

JAVA/J2EE programming language of choice was Java for this project, and different programming libraries about Java came under consideration. The best fit for the requirements of our client was found to be the HapiFHIR software library.

The Function and Code of the HL7 Data consolidation FHIR Tool

The tool was developed to fetch data from the patient's CCD, CSV files, HL7 2.x files or FHIR records, parse it into a PostgreSQL database and then make it available through Azure API gateway in a controlled environment. The Azure API is used to provide one simple FHIR API for consumers and utilize data coming from different sources and in different formats. The data are also consumed in a dashboard format using Power BI tools.

One of the primary sources of data was from JSON (JavaScript Object Notation) files from multiple resources, such as physicians, pharmaceuticals, and surgeons, etc. For JSON data parsing, the HAPI FHIR Server library came in to use. For CSV, CCD, HL7, standards Mirth Connect came underutilization as an integration engine. After parsing data into PostgreSQL database, HAPI FHIR Server a connection is made to the Azure API gateway. The output is through FHIR RESTful (Representational State Transfer) API, and thus data are served in JSON format.

www.techno-soft.com | info@techno-soft.com | (203) 676-8299

The following code snippets are provided to help readers/developers understand the parsing, processing of the data from FHIR JSON format and to utilize open source HAPI Fhir Server's library.

Following code shows the parsing of FHIR JSON data.

```
public void processResourceBundle() {
        // Create a context
        FhirContext ctx = FhirContext.forDstu3();

        // Create a JSON parser
        IParser parser = ctx.newJsonParser();
        parser.setOverrideResourceIdWithBundleEntryFullUrl(false);
        Bundle bundle = (Bundle)
ctx.newJsonParser().parseResource(resourceData);

        List<Resource> resourcesInDocument = getAllResourcesInBundle(bundle);

        processAllResources(resourcesInDocument);
    }
```

Once the data is parsed, the following code snippet shows how the different parts of the data in the JSON format is isolated into data sets that is being referred as "resource."

```
private void processAllResources(List<Resource> resources) {
        dbHelper = new DBHelper(AppConstants.DB_HOST,
AppConstants.DB_PORT, AppConstants.DB_NAME, AppConstants.DB_USER,
AppConstants.DB_PASSWORD);
        dbHelper.connect();
        try {
              for (Resource resource : resources) {
                  if (resource instanceof Organization) {
                      processOrganizationResource((Organization) resource);
                  } else if (resource instanceof Patient) {
                              processPatientResource((Patient) resource);
                  } else if (resource instanceof Condition) {
                    processConditionResource((Condition) resource);
                  } else if (resource instanceof CarePlan) {
                    processCarePlanResource((CarePlan) resource);
                  } else if (resource instanceof Observation) {
                    processObservationResource((Observation) resource);
                  } else if (resource instanceof Encounter) {
                      processEncounterResource((Encounter) resource);

                  }
                   else if (resource instanceof MedicationRequest) {
processMedicationRequestResource((MedicationRequest)resource);
                }else if (resource instanceof Claim) {
                      processClaimResource((Claim) resource);
                  } else if (resource instanceof Procedure) {
                      processProcedureResource((Procedure) resource);
                  } else if (resource instanceof Immunization) {
                      processImmunizationResource((Immunization) resource);
```

If you look carefully, you may notice a code mentioning "dbHelper." It is a Technosoft homegrown utility that reads the JSON and inserts all the relevant data in the PostgreSQL database tables. It effectively facilitates the development process and renders invaluable help to the developer for creating the database for the project that can be later leveraged by the FHIR tool to extract the patient's data.

Below is the code for dbHelper utility,

```
package com.technosoft.util;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Date;
import java.util.Map;

public class DBHelper {

	private Connection conn;
	private String host;
	private String port;
	private String dbName;
	private String user;
	private String pass;

	//we don't like this constructor
	protected DBHelper() {}

	public DBHelper(String host, String port, String dbName,
String user, String pass) {
		this.host = host;
		this.port = port;
		this.dbName = dbName;
		this.user = user;
		this.pass = pass;
	}

	public boolean connect() {
		if (host.isEmpty() || dbName.isEmpty() || user.isEmpty()
|| pass.isEmpty()) { try
			{
				throw new SQLException("Database credentials
missing");
			} catch (SQLException e) {
				// TODO Auto-generated catch
				block e.printStackTrace();
			}
		}
```


www.techno-soft.com | info@techno-soft.com | (203) 676-8299

```
		try {
			Class.forName("org.postgresql.Driver");
			String jdbcURL = "jdbc:postgresql://" + this.host
+ ":" + this.port + "/" + this.dbName;
			this.conn = DriverManager.getConnection(jdbcURL, this.user,
this.pass);
		} catch (SQLException e) {
			// TODO Auto-generated catch
			block e.printStackTrace();
		} catch (ClassNotFoundException e) {
			// TODO Auto-generated catch
			block e.printStackTrace();
		}
		return true;
	}

	public ResultSet execQuery(String query) throws SQLException {
		return this.conn.createStatement().executeQuery(query);
	}

	public int insert(String table, Map<String,
Object> values) throws SQLException {

		StringBuilder columns = new StringBuilder();
		StringBuilder vals = new StringBuilder();

		for (String col : values.keySet()) {
			columns.append(col).append(",");

			if (values.get(col) instanceof String) {

	vals.append("'").append(values.get(col)).append("',");

			} else if (values.get(col) instanceof Date) {

	vals.append("'").append(values.get(col)).append("',");

			}else {
				vals.append(values.get(col)).append(",");
			}
		}

		columns.setLength(columns.length()-1);
		vals.setLength(vals.length()-1);

		String query = String.format("INSERT INTO %s (%s)
VALUES (%s)", table, columns.toString(), vals.toString());
		System.out.println("query is:" + query);
		return this.conn.createStatement().executeUpdate(query);
	}

}
```

The following snippet shows the data relevant to the medical facility where the patient is being treated.

```
private void processOrganizationResource(Organization
organization) throws FHIRException, SQLException {
        System.out.println("Processing Organization");

        Map<String, Object> organizationRowMap =
new HashMap<String, Object>();
        organizationRowMap.put("name",organization.getName());

organizationRowMap.put("type",organization.getTypeFirstRep().getText());
        organizationRowMap.put("uid",organization.getId().substring(9));

organizationRowMap.put("address",organization.getAddressFirstRep().getText()) ;


        if (dbHelper.insert("organization", organizationRowMap) == 1)
            { System.out.println("Organization record added");
        }
```

The following snippet helps the dbHelper to insert data into the PostgreSQL database. This data consists of the different particulars of the patients, such as first name, last name, gender, DOB, Marital Status etc. All this information will be able to be retrieved to a webpage exactly according to the FHIR standard requirements.

```
/*************************************************************
     * Processing Patient Resource
     *************************************************************/
    private void processPatientResource(Patient patient) throws SQLException
{

        System.out.println("Processing Patient");
        Map<String, Object> patientRowMap = new HashMap<String,
        Object>(); patientRowMap.put("gender",
        patient.getGender().getDisplay()); patientRowMap.put("dob",
        patient.getBirthDate()); patientRowMap.put("marital_status_text",
patient.getMaritalStatus().getText());
        patientRowMap.put("marital_status_code"
        ,
patient.getMaritalStatus().getCodingFirstRep().getCode());
        patientRowMap.put("deceased",
patient.getDeceased().toString().substring(13,
patient.getDeceased().toString().length()-1));
        patientRowMap.put("patient_urn", patient.getId().substring(9));// id
starts with "uri:urn:". trim this before inserting
        patientRowMap.put("active", patient.getActive());
```


www.techno-soft.com | info@techno-soft.com | (203) 676-8299

```
        patientRowMap.put("language_text",
patient.getCommunicationFirstRep().getLanguage().getCodingFirstRep().getDispl
ay());
        patientRowMap.put("language_code",
patient.getCommunicationFirstRep().getLanguage().getCodingFirstRep().getCode( ));

        for (HumanName name: patient.getName())
            { if ( name.hasUse() &&
name.getUse().toString().equalsIgnoreCase("official") )
                { patientRowMap.put("first_name",
name.getGiven().get(0).toString());
                patientRowMap.put("last_name",
                name.getFamily()); patientRowMap.put("title",
name.getPrefix().get(0).toString());
                break;
            }
        }
        patientRowMap.put("phone",
        patient.getTelecomFirstRep().getValue()); patientRowMap.put("race",
getExtension(patient.getExtensionsByUrl(RACE_URL)));
        patientRowMap.put("ethnicity",
getExtension(patient.getExtensionsByUrl(ETHNICITY_URL)));
        patientRowMap.put("general_practitioner",
patient.getGeneralPractitionerFirstRep().getReference().substring(9))
;
        if (dbHelper.insert("patient", patientRowMap) == 1)
            { System.out.println("Patient record added");

            //TODO extract contact info to separate method and do exception
logging
            patientRowMap.clear();
            for(ContactPoint cp : patient.getTelecom()) {
                  patientRowMap.put("patient_urn",
patient.getId().substring(9));
                  patientRowMap.put("contact_point_category",
cp.getSystem().getDisplay());
                  patientRowMap.put("contact_point_type",
cp.getUse().getDisplay());
                  patientRowMap.put("contact_point_value", cp.getValue());
                  dbHelper.insert("patient_contact", patientRowMap);
            }
            //TODO extract address info to separate method and do exception
logging
            patientRowMap.clear();
            for(Address address : patient.getAddress())
                  { patientRowMap.put("patient_urn",
patient.getId().substring(9));
                  patientRowMap.put("city", address.getCity());
                  patientRowMap.put("country", address.getCountry());
                  patientRowMap.put("district", address.getDistrict());
                  patientRowMap.put("state", address.getState());

                  patientRowMap.put("postal_code", address.getPostalCode());
                  patientRowMap.put("address_line",
address.getLine().toString());
                  dbHelper.insert("patient_address", patientRowMap);
            }
        }
}
```

Following sample code snippet shows particular details about the FHIR immunization section.

```
/*************************************************************
     * Processing Immunization Resource
     *************************************************************/
    private void processImmunizationResource(Immunization
immunization) throws SQLException {
        System.out.println("Processing Immunization"); Map<String,

        Object> immunizationRowMap = new HashMap<>();

        immunizationRowMap.put("status", immunization.getStatus().toCode());
        immunizationRowMap.put("vaccine_code",
immunization.getVaccineCode().getText());
        immunizationRowMap.put("report_origin",
immunization.getReportOrigin().getText());
        immunizationRowMap.put("site", immunization.getSite().getText());
        immunizationRowMap.put("route", immunization.getRoute().getText());
        immunizationRowMap.put("practitioner_role",
immunization.getPractitionerFirstRep().getRole().getText())
        ; immunizationRowMap.put("explanation_reason",
immunization.getExplanation().getReasonFirstRep().getText());
        immunizationRowMap.put("explanation_reason_not_given",
immunization.getExplanation().getReasonNotGivenFirstRep().getText());
        immunizationRowMap.put("vaccination_protocol_target_disease",
immunization.getVaccinationProtocolFirstRep().getTargetDiseaseFirstRep().getT
ext());
        immunizationRowMap.put("vaccination_protocol_dose_status",
immunization.getVaccinationProtocolFirstRep().getDoseStatus().getText());
        immunizationRowMap.put("vaccination_protocol_dose_status_reason",

immunization.getVaccinationProtocolFirstRep().getDoseStatusReason().getText() );

        immunizationRowMap.put("lot_number", immunization.getLotNumber());
        immunizationRowMap.put("expiration_date",
immunization.getExpirationDate());
        immunizationRowMap.put("note",
immunization.getNoteFirstRep().getText());
        immunizationRowMap.put("patient_urn",
immunization.getPatient().getReference().substring(9));
 //         immunizationRowMap.put("id", System.currentTimeMillis()); if
           (dbHelper.insert("immunization", immunizationRowMap) == 1)
            { System.out.println("Immunization record added");

        }}
```


www.techno-soft.com | info@techno-soft.com | (203) 676-8299

Following code snippet shows Procedures section processing.

```
/*************************************************************
     * Processing Procedure Resource
     *************************************************************/
    private void processProcedureResource(Procedure procedure)
throws FHIRException, SQLException {
        System.out.println("Processing Procedure");

        Map<String, Object> procedureRowMap = new HashMap<>();

//        System.out.println(procedure.getNotDone()); //
no mapping column found for this
//        System.out.println(procedure.getPerformed());


        procedureRowMap.put("status", procedure.getStatus().toCode());
        procedureRowMap.put("not_done_reason",
procedure.getNotDoneReason().getText());
        procedureRowMap.put("category", procedure.getCategory().getText());
        procedureRowMap.put("code", procedure.getCode().getText());
        procedureRowMap.put("performer_role",
procedure.getPerformerFirstRep().getRole().getText());
        procedureRowMap.put("reason_code",
 procedure.getReasonCodeFirstRep().getText())
          ; procedureRowMap.put("body_site",
procedure.getBodySiteFirstRep().getText());
        procedureRowMap.put("out_come", procedure.getOutcome().getText());
        procedureRowMap.put("complication",
procedure.getComplicationFirstRep().getText());
        procedureRowMap.put("follow_up",
procedure.getFollowUpFirstRep().getText());
        procedureRowMap.put("focal_device_action",
procedure.getFocalDeviceFirstRep().getAction().getText());
        procedureRowMap.put("used_code",
procedure.getUsedCodeFirstRep().getText());
        procedureRowMap.put("performed_date_time",
procedure.hasPerformedDateTimeType() ?
procedure.getPerformedDateTimeType().getValue() : null);
         procedureRowMap.put("performed_period_start",
procedure.hasPerformedPeriod() ? procedure.getPerformedPeriod().getStart()
: null);
        procedureRowMap.put("performed_period_end",
procedure.hasPerformedPeriod() ? procedure.getPerformedPeriod().getEnd()
: null);
        procedureRowMap.put("performer",
procedure.getPerformerFirstRep().getActor().getDisplay());
        procedureRowMap.put("focal_device",
procedure.getFocalDeviceFirstRep().getManipulatedTarget().getModel())
;
        procedureRowMap.put("patient_urn",
procedure.getSubject().getReference().substring(9));

        if (dbHelper.insert("procedure", procedureRowMap) == 1)
            { System.out.println("Procedure record added");
```

 www.techno-soft.com | info@techno-soft.com | (203) 676-8299

Similarly, all the other sections, such as observations, medication, and surgery goes through a process with this tool. All this is the phase 1 one of the project that is being carried out at Technosoft.

Projects on HL7 FHIR like this one and others related with healthcare technologies have been successfully carried out at Technosoft with 99% customer satisfaction rate that numbering more than 403. Technosoft is committed to providing with top-notch services for the software development industry to develop healthcare software apps and cost-effective online collaboration to give the company's healthcare solutions that have tight integration with Information Technology and its different aspects with HL7 products like FHIR and a lot more. The software development professionals at Technosoft are well skilled in providing you with the best customer experience possible.

Just take a moment and contact us. We want to hear from you about your most challenging requirements for software development about healthcare. If you have a problem, we assure that we will come up with the best quality solution in existence.

www.techno-soft.com ☎ 203-676-8299 info@techno-soft.com