An ontology and it's realization as a Domain Specific Language on mobile devices

- a Healthcare Case study

Praseed Pai K.T.
Services and Solutions
UST Global inc
Trivandrum, Kerala
India

Shine Xavier
Services and Solutions
UST Global inc.
Trivandrum, Kerala
India

Abstract— **A leading Healthcare services company in the USA required a portable solution for eligibility determination and enrollment of indigent patients for federal, state and community benefit medical aid programs. Owing to the myriad of eligibility determination rules set by these sponsors and its volatile nature, a standard ontology based rule evaluation engine was built to target Android and iOS tablets. This Paper deals with the problem, its solutions and engineering challenges involved in realizing a computational engine from domain ontology for demographic data. By writing a JavaScript backend for an open source compiler and leveraging Google's open source JavaScript engine (V8), the authors delivered a solution which is in use predominantly in the states of Florida and Georgia (more in progress).**

***Index Terms*—Ontology, DSL, Rules,BCM, Open Source**

I. Introduction

Management of uninsured patient base is an acute problem faced by healthcare providers across the United States of America. The delivered solution helps in getting needy patients the financial help they desperately need thereby assisting hospitals in increasing cash recovery, reducing cycle time from initial patient referral to payment remittance, enhancing patient satisfaction and improving community relations.

II. THE PROBLEM

Patient Screening as part of eligibility determination is a lengthy endeavor and Healthcare representatives (HCR) travel to remote places to determine program eligibility for a potential beneficiary. Once the basic information needed for eligibility determination has been collected on paper, HCR has to travel back to his office to feed this information to an on premise application that determines the eligible programs based on pre-defined eligibility rules. Then the HCR returns with the necessary paper forms to meet the beneficiary for initiating the eligibility program enrollments. This lengthy and multi-stage process was inefficient and yielded a low throughput in terms of the number of program enrollment attempts. Each year, lot of funds earmarked for distribution by these sponsors gets lapsed because of this bottleneck.

III. PROPOSED SOLUTION

To speed up the process, UST services and solutions group recommended the creation of a mobile tablet based application to execute the entire process (from eligibility determination till program enrollment) electronically. Once program eligibility is determined and corresponding forms are filled, the contents can be synchronized with the central server (data synchronization). The solution, a simple guided forms navigation system (through a series of screening questionnaire) along with a rules evaluation engine (which runs locally in the device), automatically determines eligibility (even in disconnected mode) thus reducing the screening complexity and accelerating the program enrollments.

IV. CHALLENGES

Since the eligibility questionnaire and program enrollment forms varied with state and federal laws, there was dearth of standard terminology which was applicable across forms despite collection of demographic and social data regarding the beneficiaries. The problem of questionnaire unification arose with the presence of multiple programs within a state and the patient being eligible for more than one program. There were downstream systems and existing system infrastructure to be integrated with the proposed system. Running a rule engine on a mobile device can be a resource hog and at the time (circa 2012) authors could not find a public domain or commercial rule engine which runs on an Android or iOS device. A rule engine was necessary for eligibility determination in offline/disconnected mode.

V. Engineering a Solution

Considering the complexities involved, the project team saw the problem as an Information (knowledge) management problem and decided to have an approach different from a typical software services based solutions. The team decided to create a standard ontology for terms and concepts used in the system and got it ratified by the business. This created a shared vocabulary for all stakeholders. Entities and processes were derived from the ontology and a Domain Specific Language was created to encode business rules for determining eligibility. Due to a standard ontology across systems, building a DSL was comparatively easy.

VI. A STANDARD ONTOLOGY

Even though rooted in philosophy, Ontology [1] offers a framework for organizing information as conceptual structures and representing knowledge about the world or some part of it. They are good at describing structure, behavior and semantics of things and processes in a domain. From an information management perspective, the first task was to collect all the forms to a one central location and a group of analysts extracted data from scanned PDF documents into Excel sheet. A group consisting of Business Analysts and healthcare specialists created ontology for describing concepts embodied in the documents. Creation of a standard ontology for types, attributes and rules within system is necessary for writing a Domain Specific Language (DSL) for scripting eligibility rules.

To start the ontology development, the authors started with a well-known catalog from Jim Arlow and Illa Nuestedt[2]. The pattern catalog is based on the concept of archetypes, which was based on the work of legendary Swiss psychologist Carl Gustav Jung. Their work also emphasizes the role of Business context model and acted as a starting point of the creation of Business Context Document (BCD), for system integration.

VII. SYSTEMS INTEGRATION

Broadly speaking, Systems can be integrated at the things (entity) level or process level. For the system, Entity level integration is preferred as Entities and their relationship change slowly (of course, once it has stabilized) than business processes. We propagate changes to the entities in a structured XML based BCD document.

If we are propagating the changes at the field level (of entities), there is potential for combinatorial explosion as far as the messages are concerned. An entity consisting of 20 fields will have a possible 1048576 types of changes! This mandated the partitioning of entities into mutually exclusive set of fields called segments. **The partition of an entity should have mutual exclusivity as a constraint**.

The Change list document contains information as a top level tag called Entities. Inside Entities, details regarding each Entity changed will be encoded. Changes are recorded at the Segment level. Even If, only a field changes in a segment, the entire value of fields within a segment will be propagated. The consuming system can make a request to the web service for querying the latest patient data.

VIII. A Domain SPECIFIC LANGUAGE

Since Eligibility rules for program change intermittently, baking these rules inside the system is not a viable option. There should be a mechanism for business to specify and modify eligibility rules of the system whenever there are changes in federal or state regulation. A Microsoft Excel based template was created to key in the business rules. The Rules inside the worksheet is encoded in the syntax of SLANGFORDOTNET (http://slangfordotnet.codeplex.com) compiler. This syntax mirrors an algebraic notation and language is statically typed. One of the authors being the person who wrote the compiler, we could easily write a JavaScript backend.

The JavaScript file can be retrieved from HTTP server and can be executed by Google's V8 JavaScript execution Engine. This is the same engine which Google uses in their Chrome Browser. Incidentally, our Mobile application Development Platform was leveraging V8 for its execution. The spreadsheet contained the following worksheets. Their formats and a short description are being given here.

A. Global Variables Dictionary

VariableName	Type	Default
Insurance	bool	false
Age	int	18
Gender	string	"notspec"
Blind	bool	false
State	string	"notspec"
HI	double	0.0

B. Rule Pages

RuleName	RuleText
R1(Insurance,Age)	Return Insurance == false && Age > 18;
R2(Gender)	Return Gender == "male";

R3(Gender,Blind)	IF (Blind == true) && Gender=="male" then
	  Return true;
	ENDIF

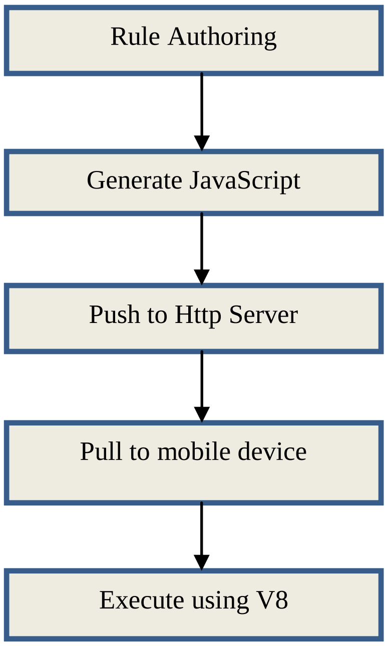
C. ProgramToRuleMapping

ProgramName	Condition
Florida_PGM	Return R1 || R3
Nevada_PGM	Return R3
FEDERAL	Return !R2 || R3

IX. Rule Pipeline

The Excel spreadsheet will be converted into a legal SLANG program and JavaScript will be generated as the output. A .NET program by the name XLS2SLANG.exe is written for the purpose. XLS2SLANG used Slang compilation Engine as an assembly (.dll) to generate JavaScript from Excel.

The rule pipeline was integrated into the system using the schema given below



X. Rule Evaluation

The generated JavaScript and API can be invoked from the application. A code snippet which shows the rule invocation from the front end program is being given below.

```
////////////
// Create an instance of
// RuleEngine... From the
// App
var rule = new RuleEvaluator();

rule_dict = {};
rule_dict["Insturance"]=false;
rule_dict["Age"]=23;
rule_dict["Gender"]=”M “;
rule_dict["Blind"]=false;
rule_dict["State"]='American';
rule.SetCurrentEnvironment(rule_dict);

//////////////////
// EvaluateAll to determine
// Elligible programs
var program_dict = rule.EvaluateAll();
for( var el in program_dict )
  console.log("Elligibility for " +
  el + "= " + program_dict[el]);

///////////////////////
// Do incremental rule evaluation
// Call ChangeFact API before
// Invoking EvaluateDelta

rule.ResetEvaluationContext();
rule.ChangeFact("Gender",”F”);
program_dict = rule.EvaluateDelta();

for( var el in program_dict )
  console.log("Elligibility for " + el +
  "= " + program_dict[el]);
```

XI. Conclusion

An enormous challenge to standardize the questionnaire across fifty states of US led the team to consider the solution based on an ontological approach to give a uniform vocabulary where all stakeholders could speak the same language about the system being built. This helped in creating a Business Context Document (a XML document) which was used to integrate downstream systems. The Rule and Eligibility engine was built based on the standard vocabulary by leveraging open source tools and compiler technology. The investment made in the project increases the throughput in terms of the number of program enrollment attempts thereby turning hospital's social burdens into cash and getting indigent patients the financial help they desperately need.

ACKNOWLEDGEMENTS

Authors wish to thank various people for their contribution to this paper; Mr. Varghese Cherian, Director, Services and Solutions, UST global for suggesting that we present this to a broader audience and Mr. Jaya Arvind Krishna , Director ,UST consulting services for encouraging us to write this. Special mention goes to Arun Narayanan, COO, UST Global , hired one of authors to work on this project. We also thank Abhilash Ponnachan, Architect, Services and Solutions, UST Global for the technical review.

REFERENCES

[1] John F. Sowa,"Knowledge Representation – Logical , Philosophical and Computational Foundations",Thomson-Boorks/Cole, pp. 51-131, 2000

[2] J.Arlow and I. Nuestadt , "Enterprise Patterns and MDA – Building Software With Archetype Patterns and UML" ,Addison Wesley , pp. 4-7,December,2003