SEMANTIC PROCESS BASED FRAMEWORK FOR REGULATORY REPORTING PROCESS MANAGEMENT

Manjula Pilaka, Fethi A. Rabhi and Madhushi Bandara

School of Computer Science and Engineering, University of New South Wales, Sydney, Australia

ABSTRACT

Regulatory processes are normally tracked by regulatory bodies in terms of monitoring safety, soundness, risk, policy and compliance. Such processes are loosely framed processes and it is a considerable challenge for data scientists and academics to extract instances of such processes from event records and analyse their characteristics e.g. if they satisfy certain process compliance requirements. Existing approaches are inadequate in dealing with the challenges as they demand both technical knowledge and domain expertise from the users. In addition, the level of abstraction provided does not extend to the concepts required by a typical data scientist or a business analyst. This paper extends a software framework which is based on a semantic data model that helps in deriving and analysing regulatory reporting processes from event repositories for complex scenarios. The key idea is in using complex business-like templates for expressing commonly used constraints associated with the definition of regulatory reporting processes and mapping these templates with those provided by an existing process definition language. The efficiency of the architecture in evaluation, compliance and impact was done by implementing a prototype using complex templates of Declare ConDec language and applying it to a case study related to process instances of Australian Company Announcements.

KEYWORDS

Regulatory Reporting, Process Extraction, Semantic Technology, Events.

1. INTRODUCTION

Regulatory Reporting Processes help regulators in monitoring the regulatory capital, safety and soundness of legal entities[1]. There are hundreds of data sources that can be used to collect different events that are pertaining to regulatory reporting such as company announcements, stock market feeds, and news data. These regulatory reporting processes are consolidation of many different sub-processes. This paper is motivated by the need to support analysts, academic researchers or data scientists who are interested in discovering important relationships between different events around specific regulatory reporting processes [2][3] which are complex in nature. The main difficulty in extracting process instances of these regulatory reporting processes is that they are loosely framed processes (there is no explicitly defined process model[4]), very complex and sometimes based on observed or inferred or looped events. Although there are many providers that offer different interfaces for analysing regulatory reporting event streams, it is still a considerable challenge to extract instances of regulatory reporting processes from these loosely framed processes, analyse certain aspects like conformance of the process instances to the process model defined and assess the impact of process instances through third-party systems and additional data. To address these issues, this paper extends a regulatory reporting process

Natarajan Meghanathan et al. (Eds) : ACITY, AIAA, DPPR, CNDC, WIMNET, WEST, ICSS - 2019
pp. 183-201, 2019. © CS & IT-CSCP 2019 DOI: 10.5121/csit.2019.91718

management framework[34] which comprises of semantic meta model and regulatory reporting process architecture. Apart from the above mentioned issues the paper is also motivated to extend the framework in leveraging existing ontologies and ensuring reproducability and consistency of framework and ability to be interoperable with other databases like Thomson Reuters.

The structure of the paper is as follows. Section 2 provides a literature review related to analysing regulatory reporting processes. Section 3 presents the proposed regulatory reporting process management framework that can be used for deriving the regulatory reporting processes. Section 4 discusses the implementation of the framework. Section 5 discusses evaluation of the framework with a case study and section 6 concludes the paper with limitations of the framework and suggestions for future work in this area.

2. BACKGROUND AND RELATED WORK

The main challenge in analysing regulatory reporting processes is that they are loosely framed processes and their process model cannot be explicitly defined[4]. Considering a process as a set of expectations, the problem is deriving a series of events that meet these expectations. Another challenge analysts face while extracting these processes is the difficulty to map different business concepts used to define rules and regulations with programming or technical concepts. A simple example is the time duration, which is expressed in business days rather than calendar days and hence needs to be adjusted for each analysis accordingly.

Here we discuss the most relevant research areas for analysing regulatory reporting processes.

2.1. Complex Event Processing(CEP)

Complex Event Processing analyses data from multiple sources to infer patterns of events that represent complex (causal, data or temporal) relationships. CEP provides event processing logic as an abstraction of event operations, separated from application logic. Some popular CEP platforms include Stream SQL[6], Oracle EPL[7], IBM Web sphere[8] and Sybase Aleri CEP [9]. The major challenge in CEP is that its implementation is highly technical and time-consuming.

2.2. Process Mining and Conformance Checking

This is a relatively new research area that focuses on exploring, observing, and improving the overall business process based on analysing event logs that record activities performed by people, software, and machines[10][11]. Process mining comprises of process discovery, conformance checking and model enhancement[12]. Process discovery is the discovery of processes from an event stream with the help of algorithms like a-priori, alpha and heuristic miner. Conformance checking means checking the processes periodically by comparing process model with process instances for any deviations or violations in processes. Model enhancement can be verifying models and proposing new models or automated construction of simulation models. Existing process definition and mining tools in the market include Disco[13], Perceptive[14] and Celonis[15]. ProM [15][16] is the largest platform available for process mining. The challenges with process mining tools are that they are not tailored to complex processes where there are conditions on inferred, observed or looped event types and require a fair amount of programming expertise with deep technical knowledge to implement rules.

2.3. Process Definition and Modelling Languages

There are appropriate process modelling systems for loosely framed processes such as Flower (Pallas Athen case handling system)[17] and Tibco Inconcert (an ad-hoc workflow system)[33].

In a case handling system, a process is a recipe for handling cases of any given event type together with activities, which are the logical units of work to be executed when handling a case. Tibco Inconcert is a workflow system designed for unstructured processes and its specialty lies in creating new process models from existing cases using templates that can be adapted to a new case or model a process while executing[18]. Another well-known process modelling language is XML based Declare ConDec[19] which allows process extraction rules to be defined with the help of constraint templates. Each template has linear temporal logic specifying the semantics and a graphical representation [20]. Declare ConDec is a constraint-based Process Mining Relational language that allows for definition, verification, execution of constrained based process models and ad-hoc change of process instances. The major advantage of Declare is the flexibility to design, change, specification of complex process models. These approaches are closely related to our work in extracting process instances of complex processes but limited in providing only technical abstractions for the analysts to express the rules related to process instance extraction.

In conclusion, existing approaches do not deal adequately with the challenges as they demand both technical knowledge and domain expertise from the users. In addition, the level of abstraction provided does not extend to the concepts required by a typical business analyst. An ideal solution would need to use both capabilities of complex event processing and process analysis technologies under the same framework in a complementary way.

3. PROPOSED ARCHITECTURE FOR REGULATORY PROCESS MANAGEMENT FRAMEWORK(RPMF)

In this section, we propose an overview of proposed Regulatory Process Management Framework. In Section 3.1, we explain the semantic data model in detail by referring to each of its constituent ontologies. In section 3.2, we describe the Regulatory Process Management System Architecture and the proposed main algorithm of the architecture - process extractor algorithm and section 3.3 concludes the section.

3.1. Semantic Data Model

The proposed semantic data model provides flexibility to represent high level concepts related to regulatory reporting. These high level concepts are originating from different ontologies.

- Event: identifying a wide range of events related to the regulatory reporting domain such as company announcements, financial market data etc.
- Domain: identifying several concepts related to the underlying domain.
- Process: identifying number of concepts representing regulatory processes.

3.1.1. Event Ontology

The event ontology described in this section is based on Adage event meta modelling framework[21] which has been specifically designed for event data analysis. The Adage event data modelling framework provides an event data meta model with a set of operational guidelines to help create event data models irrespective of sources or domains[22]. Here are some of the main concepts that are represented in our event ontology.

1. *Event* – A super class which could represent any kind of event that has a timestamp
2. *FinancialMarketEvent* – A subclass of Event which relates to a financial instrument
3. *FinancialInstrumentID* – Defines the instrumentID, for example a company announcement, has an instrumentID which corresponds to a company code
4. *Eventattribute* – Super property of the event and any event attribute used in semantic data model is a sub property of this concept

5. *Company Announcement* –A subclass of FinancialMarketEvent which represents the release of a market announcement. A company announcement event has multiple attributes for example; headline of the announcement, category code and price sensitivity (whether the announcement is likely to impact the stock price significantly).
6. *End of day event* – A subclass of FinancialMarketEvent which represents a summary of trades of stocks of a company daily (e.g. closing price).

3.1.2. Domain Ontology

Semantic interoperability enables not just the packaging of syntactical data but also simultaneous transmission of unambiguous meaning of data through metadata and linking all the metadata through a shared vocabulary. Event attributes refer to domain concepts, so there is a need for an ontology that represents these concepts in a standardised way. For example, the Financial Industry Business Ontology (FIBO)[23] is an initiative to define financial industry terms, definitions and synonyms using semantic web principles and Object Management Group (OMG) modelling standards.

Here are some of the concepts that are used in our Domain Ontology from FIBO.

1. *Date time* – A date identifies a calendar day on some calendar. Datetime is a combination of date and time without a time zone.
2. *Duration* – Amount of time between the 2 events.
3. *BusinessDayConvention* – BusinessDayConvention is the number of possible ways in which a date that falls on a weekend or on a holiday can be handled and BusinessDayTreatment combines the date with Businesscenter to determine the course of action when a business is conducted on a specific business day in a business centre.

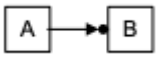
3.1.3. Process Ontology

To represent regulatory reporting processes in a flexible way, we adopt the approach used in loosely framed process languages which are based on the principle of defining processes as a set of constraints involving events and event entity conditions. Our process model will support several templates specifically adapted to regulatory reporting.

The proposed concepts are:

- *RPMSProcessType*: uniquely defines a regulatory process with its constraints and associated templates and parameters.
- *RPMSConstraint*: represents the application of RPMSTemplate with its parameters
- *RPMSTemplate*: represents a common set of parameters involving events and/or domain concepts. Such RPMS templates are implemented on top of Declare ConDec [19] templates. In other words, our templates can be considered as an application of Declare templates customised for the Regulatory Process Modelling domain.
- *RPMSParameters*: provides links to the actual parameters of the template which will be instance events and/or domain concepts. The list of RPMS templates with the extension of previous semantic meta model with chain precedence template and their expected parameters is shown in Table 1.

Table 1. Three types of RPMS templates and respective parameters

RPMS Templates	RPMS Parameters	Role and Function	Notation of Relation Templates
Precedence	Event1(A), Event2(B), Duration	Determine if the business day duration between two events is within parameter duration	
Alternate Precedence	Event1(A), Event2(B), Duration	Determine if the alternate business day duration between two events is within parameter duration	
Chain Precedence	Event1(A), Event2(B), Duration1, Duration2	Determine if the business day Duration1 between Event1 and Event2 is within parameter duration and if the business day Duration2 between Event 2 and Event 1 is within the parameter duration	

3.1.4. Proposed Ontology - Semantic Data Model

Figure1 shows integrated RPMS Semantic data model that includes all underlying ontologies. We can see that the process modelling constraints or conditions can refer to business concepts. Events can be any financial events like ASX announcements, End of Day etc. Business concepts refer to FIBO classes for example business day duration which helps derive the constraint verification based on business day conventions. For example: if we consider Precedence Template in Table 1, Event 1 and Event 2 can be ASX announcements and the Duration can be a FIBO duration in calendar days as per business day convention.

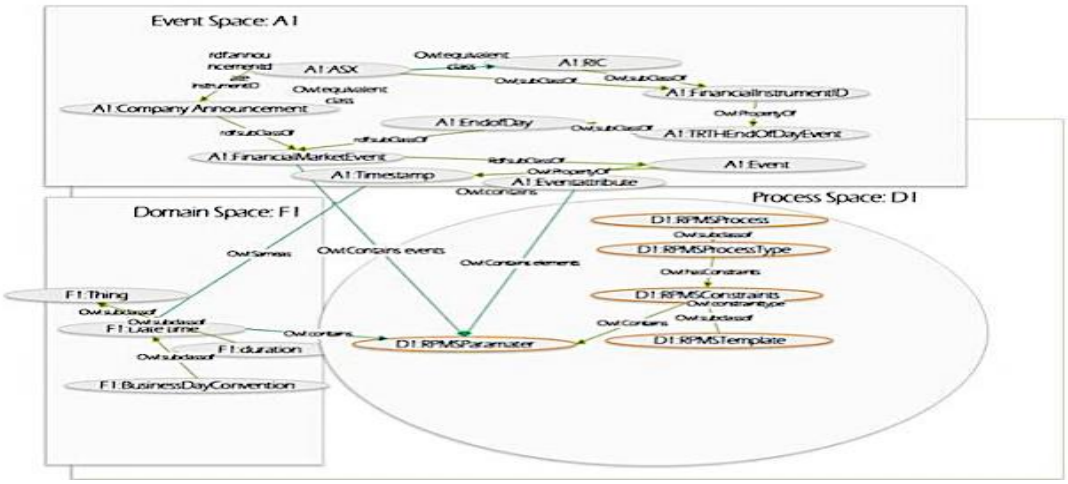


Figure 1. RPMS Semantic Data Model

3.2. Proposed Regulatory Process Management System Architecture

This section details the proposed architecture with its overall structure and describes how RPMS templates are implemented and explains the main algorithm which extracts process instances from event instances stored in the semantic database.

3.2.1. Overall Architecture

The architecture supports several analysts in an organisation in managing knowledge related to regulatory reporting. The important roles supported by the architecture are

- *Process Definer*: defines processes and constraints for a specific process
- *Model Maintainer*: maintains model – eg: adding a new event model or making extensions to the existing meta model
- *Analyst*: responsible for maintaining event data in Semantic database and performing conformance and further analysis on the data with the help of RPMS Website.

The proposed architecture shown in Figure 2 is a five layered architecture. The UI Layer has 3 GUIs. The first one is the Semantic Model Maintainer GUI which is used to create or maintain the semantic event model. The second one is the Process Definer GUI which is used to define process types with the sequence of events comprising process types and the constraints between the events. The third one is the RPMS (Regulatory Process Management System) website for importing events, extracting processes and analyzing processes.

The Business Logic Layer has 5 main modules:

- *Process Definer* module builds the process type definition file via Process Definer GUI.
- *Event Importer* module converts an event data file to RDF format (triples) and helps store the data in the Semantic database.
- *Meta Model Maintainer* module helps in accessing, updating and maintaining the semantic meta model.
- *Process Extractor* Module is the main module which extracts the process instances from the semantic database with the help of the process type definition file. The Process extractor algorithm is detailed below in this section.
- *Process Analyzer* helps in analysing the extracted process instances further and provides details of the analysis to be visualized with SPMS website.

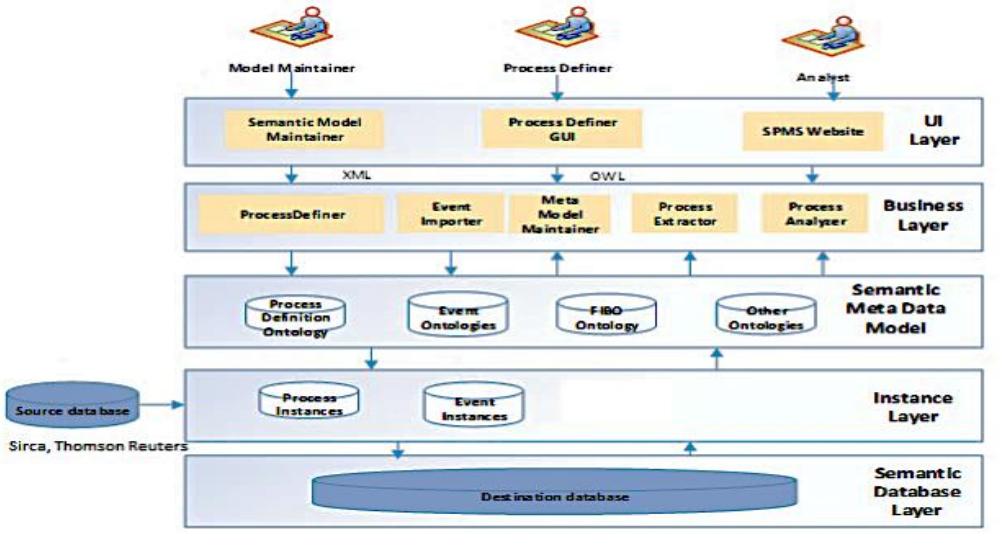


Figure 2. Architecture for Regulatory Process Management System

The Semantic Data Meta Model Layer defines the meta model using the underlying ontologies. The Instance Layer stores the process, event and other object instances which align with the semantic data model. The Instance layer relies on a Semantic database layer which physically

stores all objects required for the Regulatory Process Management Framework. In the next section, we describe the RPMS templates implementation in detail.

3.2.2. RPMS Templates Implementation

RPMS templates are built on top of Declare ConDec[19] templates which expect different concepts as parameters. Declare has more than 20 constraint templates which can be categorised into four categories Existence, Relation, Choice and Negation templates. A constraint inherits name, graphical representation and the linear temporal logic (LTL) formulas with each of these templates. Each of these constraints consists of conditions, constraint parameters and templates with graphical representation and state messages. During execution LTL formulas are applied on activities which are provided as arguments. All the data model elements of activities and constraints are mapped to data model elements of the model which are declared globally in the model. The mappings between different RPMS process concepts and Declare template concepts are illustrated in Table 2.

Table 2. Mapping of RPMS Templates with Declare Templates

RPMS Template	**Declare Template**	**Declare Template Parameters**
Precedence	Precedence	Branches (Activity1, Activity2), condition(Duration), LTL Logic, Data Elements
Alternate Precedence	Alternate Precedence	Branches (Activity1, Activity2), condition(Duration), LTL Logic, Data Elements
Chain Precedence	Chain Precedence	Branches (Activity1, Activity2),condition(Duration) LTL Logic, Data Elements [Activity1 precedes Activity2 and Activity2 precedes Activity1 in a loop if condition(Duration) is satisfied]

An example of RPMS Precedence template is shown in Figure 3. Constraint1 has P1 Precedence template with activities A1 and A2 with a condition C1 between them where A1 is the Event Type category code with value = “1001” and A2 is the Event Type category code with value = “1002” and C1 is the condition with the difference between A1 and A2 in Business day duration.

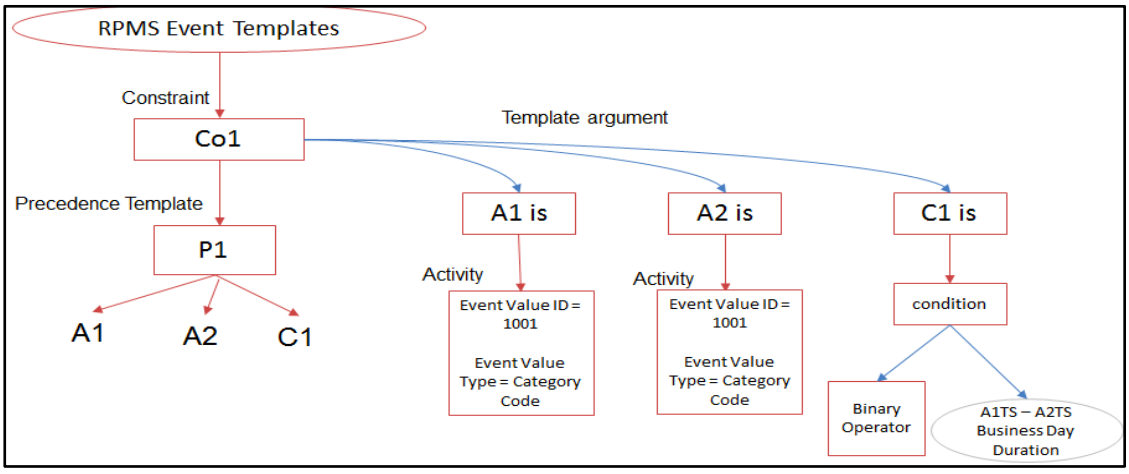


Figure 3. Regulatory Process Management System(RPMS Precedence Template)

3.2.3. Process Extraction Algorithm

The role of the process extraction module is to extract process instances from event instances. This is the main component of the proposed system and is shown in Figure 4.

```
#Detecting all instances of a process type Algorithm ProcessExtractor
1.Input: RPMS ProcessType(P)
2.Output: Event instances for the specified process type(P)
3.Query set of all constraints associated with RPMS Process Type P -> {RPMS
Constraint C} 4.For each RPMS constraint, C
5.Extract RPMS template(T) associated with C with RPMS template parameters p1, p2,.
pn 6.Extract associated values of parameters ie., v1..vn
7.Retrieve Declare template associated with T
8.Connect v1..vn with Declare template parameters
9.Extract code (Linear Temporal Logic or LTL logic) associated with the declare
template 10.Execute code with corresponding Declare template parameters
11. Store the result set for each constraint C
12.Return result data retrieved after merging results for all constraints
```

Figure 4. Process Extractor Algorithm

4. IMPLEMENTATION OF REGULATORY PROCESS MANAGEMENT FRAMEWORK(RPMF)

This section describes a prototype implementation of the proposed Regulatory Process Management Framework to validate its feasibility and functionality.

4.1. Semantic Meta Model, Instance, Database layer

This section provides details on how the semantic model proposed in section 3 was implemented using realistic data.

4.1.1. Data Sets

In this implementation two datasets are used to create instances of the Event Ontology. These datasets have been acquired from Thomson Reuters and Sirca (http://www.sirca.org.au), two prominent financial data repositories. First dataset is the Australian Company Announcements (ACA) which holds historical records of announcements and provides extensive search capability for researchers. The second dataset is the TRTH which contains high frequency data such as quotes, trades, market depth occurrences and low frequency data like end of day prices. Both these datasets organise data in the form of textual files in csv format, and are accessible via web portals.

4.1.2. Mapping Data with the Semantic Meta Model

For this implementation, we are using two entities available in datasets: end of day event and announcements. Each entity is mapped to the semantic meta model class using the Protégé tool [24]. Table 3 shows the mapping of dataset entities with their corresponding semantic meta model classes and the attributes of these entities that have been used in this implementation.

Table 3. Mapping of event ontology with Semantic Meta Model classes and attributes

Entity	**Semantic Meta Model Class**	**Attributes of each of the Semantic Meta Model classes**	**TRTH or ACA attribute**
End of Day	End Of Day Event (subclass of Financial Market Event)	Open price Close price Eod Volume EOD _Date EOD _Instrument ID	Open Last Volume Date #RIC
Announcements	Company Announcement (subclass of Financial Market Event)	Instrument ID Announcement Date Announcement Time Announcement Category Announcement Headline Price Sensitive	Company Date Time Category Headline Price Sensitive

In this implementation, the FIBO Business dates ontology [25] has been obtained from OMG website and used in the Domain Ontology. FIBO Duration and date entities are integrated into the semantic meta model as Duration and Datetime classes. Table 4 shows the how the entities in datasets are mapped with the semantic meta model classes and FIBO using protégé tool.

Table 4. Mapping of domain ontology with Semantic Meta Model classes and attributes

Entity	**Semantic Meta Model Class**	**Attributes of each of the Semantic Meta Model classes**	**FIBO (Business Dates.rdf)**
Duration, date time	Duration, Date time	Duration Date time	Duration Date time

This implementation uses the Process Ontology from Declare Designer[19] which stores activities and entities in an xml format. Declare has many entities and attributes but only the entities and attributes shown in Table 5 are mapped to semantic meta model classes and attributes used in this implementation.

Table 5. Mapping of process ontology with Semantic Meta Model classes and attributes

Semantic Meta Model Class	**Attributes of each of the Semantic Meta Model classes**	**XML tag from Declare**
RPMS Process Type	RPMS Process Type Name	name
Constraint (subclass of RPMS Constraints)	Condition Constraint ID Constraint Type Constraint Branches Constraint Data Element	Condition Id Mandatory Branch name Data element name
Template (subclass of RPMS Process Template)	Template Name Template Text Template Description	Name Text Description
State Message (subclass of Template class)	Message Text Message State	Html state
Date Element (subclass of RPMS Parameters)	Data Element ID Data Element Name Data Element Type Initial Value	Id Name Type initial
Activity	Activity ID Activity Name Activity Data Element	Id Name Data element

4.1.3. Instance Layer

The instance layer consists of instances from the data sets which are converted to RDF triples format using the **"**Convert Event Instance Data" algorithm. Table 6 provides the mapping of meta model entities with the source and converted file formats of each of the ontologies in the implementation.

Table 6. Event, Domain and Process Meta Model entities with file formats

Data entities from Event, Domain and Process Ontologies	**Data type**	**Source File Format**	**Converted File format**
End of Day	TRTH format	CSV file	RDF file
Company Announcements Metadata(Summary)	Sirca format	CSV file	RDF file
Business Dates Ontology	Ontology	RDF file	RDF file
Declare	Declare format	Xml file	RDF file

4.1.4. Database Layer

A Mark logic database[26] has been chosen for storing the Semantic event instance data, process definition and meta model files. Mark logic is a NOSQL database that has evolved to store documents and RDF triples. They provide query interfaces to run queries on the linked data with

SPARQL, X Query, JavaScript or SQL languages.

4.2. Business Layer Implementation

The Business layer components have been developed using Python and Django libraries [27]. We now describe each of the components described in section 3.2.1 in more detail.

4.2.1. Process Definer Module

Process Definer module builds the process type definition file via Process Definer GUI. This is used to create the process definitions from a process definition file in xml format created by the Process Definer GUI. Process types are identified from the process definition file and converted to RPMS templates according to Process Definition Ontology. Figure 5 shows the algorithm used by the module for converting Process Instance data to triples in RDF or turtle format. This data is then stored in the Process Instances database.

#Converting process instance data to triples Algorithm Convert Process Instance Data to RPMS Process Type
1. Input: Process type with event instance data in xml format
2.Output: RDF file with triples of RPMS Process Type (P) instance data
3.Write Prefixes for namespace and database in Semantic database
4.Create a file in RDF or turtle(ttl) file
5.Create rdf: type owl: Named Individual string RPMS Process Type(P)
6.For each of the Constraints in dataset C{c1..cn}
7.Extract Activities{a1..an} and Data elements{d1..dn} for each of Constraint C
8.Set data types of Activities and associated data elements for C
9.Extract Declare template(T) associated with C with template parameters p1, p2,. pn 10.Extract associated values of template parameters ie., v1..vn
11.Extract LTL logic associated with template T
12. set data types of template T with template parameters{p1..pn} and associated values v1..vn with LTL logic
13.Write string to ttl file
14.Save ttl file to Process Instance database

Figure 5. Convert Process Instance Data to RPMS Process Type Algorithm

4.2.2. Event Importer Module

Event Importer module converts an event data file to RDF format (triples) and helps store the data in the Semantic database. Figure 6 shows the “Convert Event Instance Data to Tripples” algorithm used for converting event data to semantic triples.

#Converting event instance data to triples Algorithm ConvertEventInstanceDatatoTripples 1.Input: event instance data in csv format
2.Output: triples of event instance data
3.Write Prefixes for namespace and database in Semantic database
4.Create a file in RDF or turtle(ttl) file
5.For each of the event instances in dataset E{e1..en}
6.Get all the corresponding values v1..vn for e
7.Set data types of Event e to match the schema of database
8.Create rdf: type owl: NamedIndividual string with all its attributes {a1..an} for event e 9.Write string to ttl file
10.Save ttl file to Semantic database

Figure 6. ConvertEventInstanceDatatoTripples Algorithm

4.2.3. Meta Model Maintainer

Meta Model Maintainer Module is used to maintain the semantic meta model according the actions of the Semantic Model Maintainer GUI. This GUI (Protégé editor[24]) helps in adding and extending classes in the Process Ontology and Event Ontology.

4.2.4. Process Extractor Module

Process Extractor Module is the most complex module of the implementation and an important one. It assumes that Process and Event Instance data is already stored in Process and Event Instance databases. Given a Process Type (P) as input, the Process Extractor Module implements the ProcessExtractor algorithm. The algorithm helps get the associated RPMSConstraints and Templates associated with each of the Constraints with the template parameters and the corresponding linear temporal logic code (LTL Logic) from the ProcessInstance database. The RPMSConstraints of the RPMSProcessType are then applied on the Event Instance data obtained from Event Instance database. For the RPMSConstraints based on FIBO Business dates element (Duration), a FIBO Duration Sub Module has been developed which takes Startdate and enddate as input and provides duration in Business Days (BusinessDayDuration). With the help of BusinessDayDuration the conditions are then applied on the event instance data and ProcessType event instances are retrieved for the ProcessType (P).

4.2.5. Process Analyser Module

The Process Analyser Module processes the process instance data of Process Type (P) produced by ProcessExtractor Module with EndOfDay data obtained from Thomson Reuters database. End of the day data contains the Stock price and stock volume details of a company code for which the process type is being extracted with finer details like the Stock price (end of the day), stock price (beginning of the day), the average stock price etc. Using this module the Stock Prices and Stock Volume Variations for each company are retrieved and integrated with Process Instance data for the chosen Process Type. The results are shown using the SPMS Website.

4.3. User Interface Implementation

The section describes the user interfaces used in the prototype to design the semantic model, define and extract the processes and do further analysis.

4.3.1. Semantic Model Maintainer

For implementing Semantic Model Maintainer, the Protégé editor [24] was selected for adding new event entities or maintaining existing models. An open source ontology editor and framework, Protégé was used to integrate the entities from the datasets. Using Protégé the three ontologies are combined and a semantic meta model is created. All the entities and associated attributes are added and mapped and stored in rdf or turtle format. This file is saved in local directory. The rdf file from the local directory is then saved to the semantic database with the help of the SPMS Web site Settings form.

4.3.2. Process Definer Graphical User Interface

Declare designer [32] adheres to a declarative workflow modelling paradigm while providing flexibility. The Process Definer GUI (Declare designer User Interface) is used in extending Declare ConDec language with features like declare activities, templates, constraints, conditions for semantic meta model. The Process Definer GUI first creates process instances which can be saved as an XML file in local directory. Using SPMS Website Settings form the file is then converted to Process Instances (RDF format) by invoking the Process Definer Module (see Section 4.2.1) from the Business layer and uploaded to MarkLogic server and then saved in MarkLogic database.

4.3.3. Semantic Process Management System Web Application

Semantic Process Management System Website shown in Figure 7 facilitates following activities:

- Home Page provides detailed steps involved in the process.
- Settings Page uploads the process definer, semantic meta model with process and event instances data to Marklogic server.
- Conformance Page gets the process type instance data for the company chosen from the Marklogic database.
- StockPriceImpact and StockVolumeImpact Pages show the impact of Stock Prices and Stock Volume for the selected company using the process type instance data extracted with the “Conformance” Page.

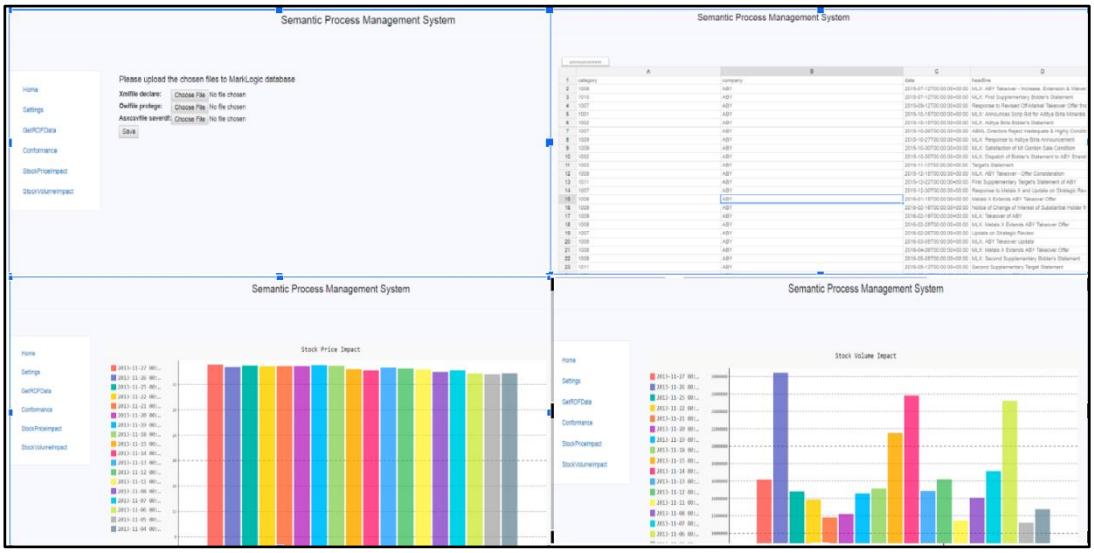


Figure 7. Semantic Process Management System Prototype

5. EVALUATION

The Regulatory Process Management Framework is evaluated using a case study on off-market bid regulatory process[34] and is being extended in this paper to evaluate with a case study of analysing quarterly reports regulatory process of Australian Stock Exchange announcements. We evaluated three objectives of the framework as shown in Table 7.

Table 7. Research objectives of case study

Objectives	**Case Study Quarterly reports process**
Leverage existing ontologies/ Extensibility	Ability to define a Process type with Semantic Model Maintainer and Process Definer GUI in a complex scenario with existing ontologies like FIBO and Adage.
Reproducibility/ Consistency	Ability to define and extract process instances of a process type with Complex Declare templates like Precedence, Alternate Precedence and Chain Precedence using "Sirca Australian company announcements" database.
Interoperability	Ability to import data and extract process instances and perform impact analysis using other data sources like Thomson Reuters.

5.1. Overview of Quarterly Reports Regulatory Process

To provide an understanding of quarterly reports regulatory process, we present Interim Financial Reporting (IFR). IFR can play an important informative role in capital markets, provided the contents of the interim reports are accurate and timely. An Interim financial report is a financial report that contains a condensed set of financial statements for a period shorter than the full financial year [29].

Process for Quarterly activity report with an indicative time table is shown in Figure 8

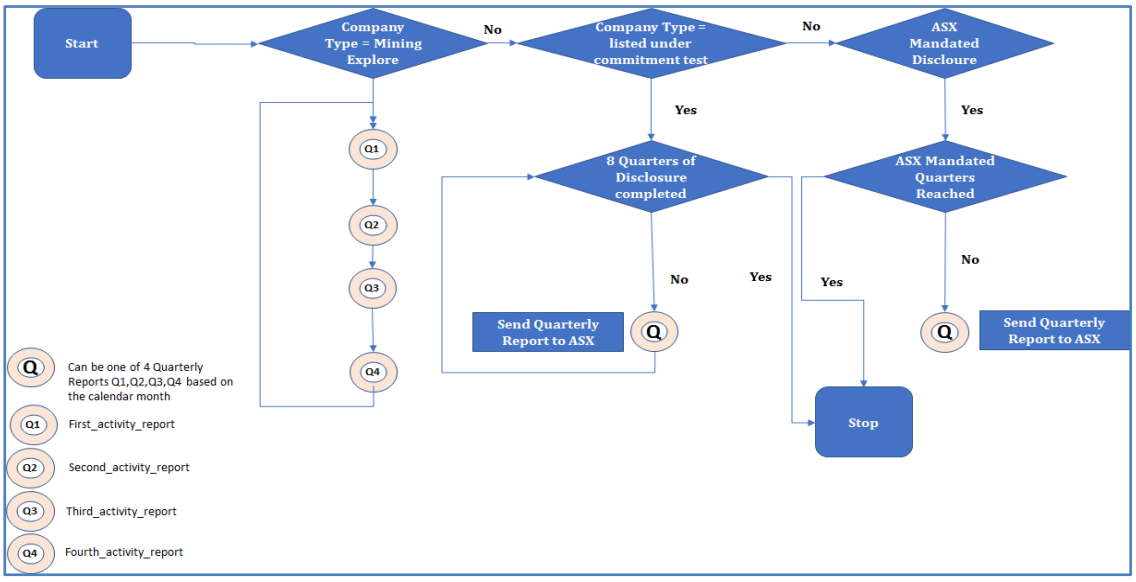


Figure 8. Regulatory Process with indicative timetable for Quarterly Reports

Rules for the process are summarised in Table 8

Table 8. Rules derived for the Quarterly activity reports process type

Company Type	**Quarterly activity report**
Mining Explore	Send First, Second, Third and Fourth activity report.
Companies listed under commitments test in ASX	Send for 8 quarters from the quarter the company is listed under commitments test in Australian Stock Exchange. Send the quarterly activity report from the quarter it has been listed for 8 quarters.
companies not listed under commitments test but mandated by ASX to share the quarterly report(Number of quarters disclosed to company by ASX)	Send the quarterly report from the quarter it has been asked to share the quarter report by ASX till the ASX mandated quarters limit is reached.

5.2. Implementation of case study

The Quarterly activity reports case study causes the semantic meta model to be extended with the QuarterlyActivityReport event. The Metamodel extension of Quarterly activity report is then viewed with the help of Protégé plugins - Onto Graph [31]. Event types of Quarterly activity reports regulatory process are listed in Table 9.

Table 9. Event types of quarterly activity reports

No	**Event Types of this case study**	**Nature of the event**
1	First_activity_report(Financial Year)	Observed event(ASX 92 Announcement Category Code = 4001)
2	Second_activity_report(Financial Year)	Observed event(ASX Announcement Category Code = 4002)
3	Third_activity_report(Financial Year)	Observed event(ASX Announcement Category Code = 4003)
4	Fourth_activity_report(Financial Year)	Observed event(ASX Announcement Category Code = 4004)

Three RPMS Templates Precedence, Alternate Precedence and Chain Precedence templates are used as shown in Figure 9

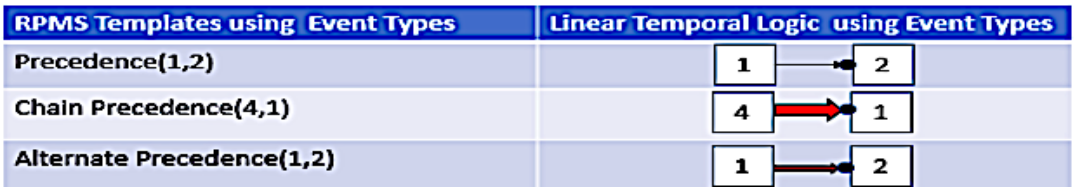


Figure 9. Three RPMS templates precedence, alternate precedence and chain precedence used for quarterly activity reports case study

The complete Process Type defined using the Process Definer GUI is shown in Figure 10.

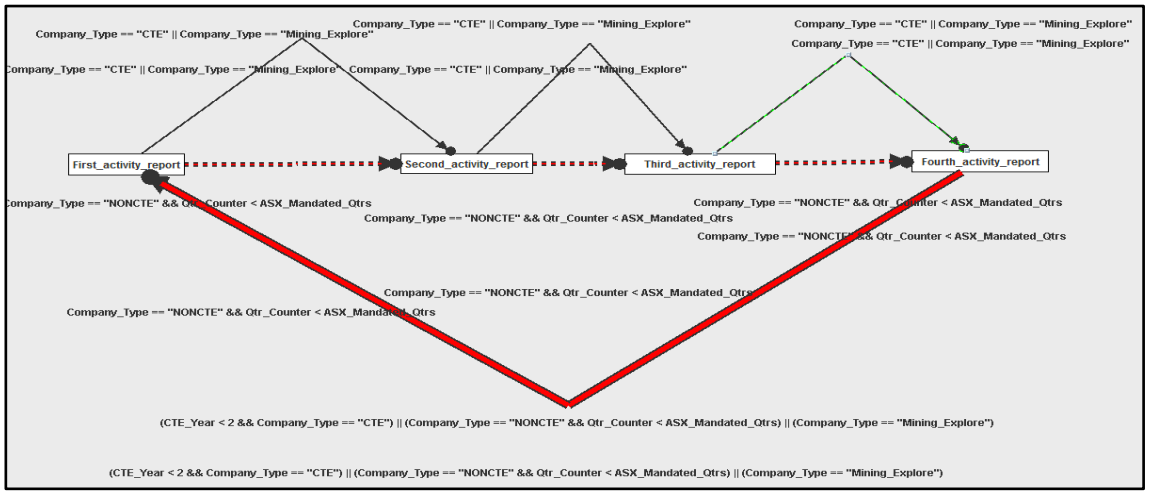


Figure 10. Quarterly activity reports process constraints defined with Declare Designer

The steps used by the user to store and extract the Quarterly activity report process type and instance data and analyse the process instance data extracted using SPMS website are:

1. Using Settings Page as shown in figure 7, the user uploads the process definer file obtained from Process Definer GUI (xml file), the semantic meta model file from Semantic Metamodel (ttl file) and the Quarterly activity reports event instance data (csv file) after conversion to rdf to user working space (i.e. Marklogic server). During the process of uploading, the data in the files are converted to instance data.

2. Using the Conformance Page, the Process Instance data for the Quarterly activity report process type is retrieved with the help of constraints defined in Process Definition file.

3. Using the Process Analyser Module, the StockPriceImpact and StockVolumeImpact Pages of the SPMS Website, End Of Day data is analysed by considering the dates of the process instances of the Quarterly activity reports process. Using machine learning techniques, this data is evaluated to retrieve the trend of the stock prices and volume during the quarterly activity reports regulatory process.

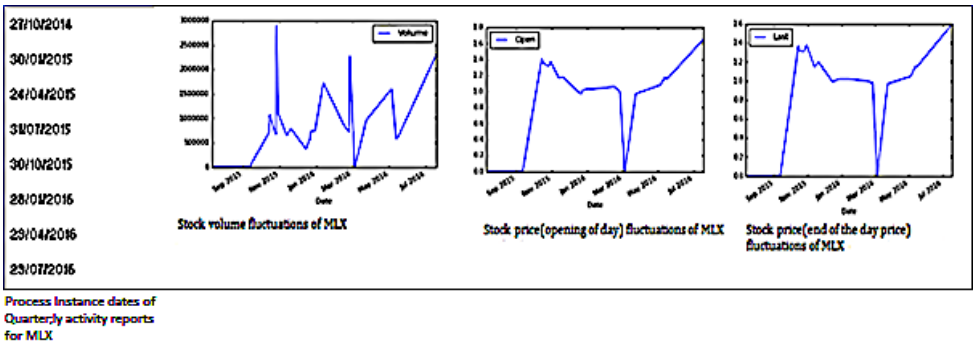


Figure 11. Stock Price and Volume fluctuations during MLX quarterly activity report process

As an example, process instance data for MLX company for quarterly activity report process has been extracted and evaluated with the stock price information from Thomson Reuters EndOfDay dataset. The following results have been observed as shown in figure 11. It has been observed that

the Stock price data (opening of the day) is slightly different from the Stock price data (End of day). It has also been observed that Stock Price fluctuations have been more compared to Stock Volume fluctuations for MLX and during the Quarterly activity reports process.

6. CONCLUSION

6.1. Discussion

This work is motivated by the challenges faced by the data scientists or academic researchers in deriving regulatory process instances. It extends the previously proposed semantic meta model within a Regulatory Process Management Framework (RPMF). The case study has revealed that it was possible to integrate existing domain, process and event ontologies like FIBO, Declare and Adage for complex scenarios of quarterly activity report which are looped or recursive. The Adage ontology was extended with ASX Announcements and the Process Ontology is extended with the complex RPMS templates like chain precedence. However, in both cases, a programmer was needed to do that task. The case study has revealed that the templates were used to model rules for recursive quarterly activity, but the user still needed semantic modelling experience to be able to supply parameters for the templates. The case study revealed that the framework is flexible enough to be integrated with existing databases like Sirca, ASX announcements and Thomson Reuters.

6.2. Limitations

The framework is focused on leveraging existing ontologies into a unified semantic model by integrating Domain, Process and Event Ontologies. As the case study only focuses on Sirca and Thomson Reuters data sources, the framework was quite dependent on them. Hence there is a possibility that the framework could have missed important event types that are present in other databases. The implemented algorithms support the RPMS framework, but an IT expert is still required to implement any changes to Semantic Model. The framework is also designed to be consistent and reproducible with any process type. Future work is required to evaluate the framework with other process types and eliminate the need for programming skills for any changes to the system. The framework requires a process designer and semantic model maintainer to design the process types using Process Designer GUI and create and maintain semantic model with the Semantic Model Maintainer GUI.

6.3. Future work

As the framework has been evaluated with only one case study future work is required in extending the work to other domains and process ontologies. Although the case study is focused on ASX announcements and EndOfDay data, this framework can be leveraged to other types of financial events like Trade, Quote, News and other data sources like Bloomberg and Raven pack which should be explored as part of future research.

REFERENCES

[1] Stewart, J(2004): American Banker- Price of Increased Regulatory Burden: Less Time for Customers,https://www.americanbanker.com/news/price-of-increased-regulatory-burden-less-time-for-customers

[2] Harrington, W., Heinzerling, L., Morgenstern, R. D., (Eds.): Reforming regulatory impact analysis. Routledge. (2010)

[3] Carroll, P.: Regulatory Impact Analysis: promise and reality. In: The ECPR/CRI Conference 'Frontiers of Regulation. Assessing Scholarly Debates and Policy Challenges', University of Bath. (2006)

[4] Dumas, M., Van der Aalst, W. M., Ter Hofstede, A. H.: Process-aware information systems: bridging people and software through process technology. John Wiley & Sons. (2005)

[5] Riyanto, A.: The Regulation of Takeovers in Australia. Law Review 5.3. (2013)

[6] Jain, Namit, et al. "Towards a streaming SQL standard." *Proceedings of the VLDB Endowment* 1.2 (2008): 1379-1390

[7] Zang, C., Fan, Y.: Complex event processing in enterprise information systems based on RFID. In: Enterprise Information Systems 1.1, pp.3-23. (2007)

[8] Marechaux, J.L.: Combining service-oriented architecture and event-driven architecture using an enterprise service bus. In: IBM developer works, pp. 1269-1275. (2006)

[9] Etzion, O., Magid, Y., Rabinovich, E., Skarbovsky, I., Zolotorevsky, N.: Context aware computing and its utilization in event-based systems. In: Proceedings of the Fourth ACM International Conference on Distributed Event-Based Systems. ACM. (2010)

[10] van der Aalst, W.M.P., La Rosa, M., Santoro F.M., Special Issue on BPM "Use Cases", Business & Information Systems Engineering, Vol. 58 No. 1, (2016)

[11] van Der Aalst, W., Adriansyah, A., De Medeiros, A.K.A., Arcieri, F., Baier, Blickle, T., Bose, J.C., van den Brand, P., Brandtjen, R., Buijs, J., Burattin, A., Process mining manifesto. In: Business process management workshops. Springer Berlin Heidelberg. (2011)

[12] van Der Aalst, W.: Process mining: discovery, conformance and enhancement of business processes. Springer Science & Business Media, 2011

[13] Schmitt, B.: Process mining with Process Observer and Fluxicon Disco. SAP Community Network (2014)

[14] van der Aalst, W. M.: Process mining: data science in action. Springer, (2016)

[15] van der Aalst, W. M., Alfredo B., van Zest S. J.: Rapid Prom: mine your processes and not just your data. (2017)

[16] van der Aalst, W. M.: Exploring the CSCW spectrum using process mining. In: Advanced Engineering Informatics 21.2, pp. 191-199. (2007)

[17] van der Aalst, W. M., Weske M., Grünbauer D.: Case handling: a new paradigm for business process support. In: Data & Knowledge Engineering 53.2 pp.129-162. (2005)

[18] Tzeremes, V., Gomaa H.: XANA: an end user software product line framework for smart spaces. In: 2016 49th Hawaii International Conference on System Sciences (HICSS). IEEE, (2016)

[19] Pesic, M., Schonenberg H., Van der Aalst M. P.: Declare: Full support for loosely-structured processes. In: Enterprise Distributed Object Computing Conference, 2007. EDOC 2007. 11th IEEE International. IEEE. (2007)

[20] Pesic, M., Schonenberg, M. H., Sidorova, N., van der Aalst, W. M.: Constraint-based workflow models: Change made easy. In: OTM Confederated International Conferences on the Move to Meaningful Internet Systems. Springer Berlin Heidelberg, (2007)

[21] F.A. Rabhi, A, Guabtni and L. Yao, A Data Model for Processing Financial Market and News Data, *International Journal of Electronic Finance*, Vol 3 (4), 2009, pp. 387-403, ISSN: 1746 0069

[22] L. Yao and F.A. Rabhi, Building Architectures for Data-Intensive Science Using the ADAGE Framework, *Concurrency and Computation: Practice and Experience*, 27(5), April 2015, pp. 1188-1206

[23] Newman, David. "The Financial Industry Business Ontology." (2013)

[24] Noy, Natalya F., et al. "Creating semantic web contents with protege-2000." *IEEE intelligent systems* 16.2 (2001): 60-71

[25] Newman, David. "The Financial Industry Business Ontology." (2013)

[26] McCreary, Dan, and Ann Kelly. "Making sense of NoSQL." *Shelter Island: Manning* (2014): 19-20

[27] Forcier, Jeff, Paul Bissex, and Wesley J. Chun. Python web development with Django. Addison-Wesley Professional, 2008

[29] Cuong, Nguyen Huu, Gerry Gallery, and Tracy Artiach "Interim financial reporting in the Asia-Pacific region: a review of regulatory requirements." Corporate Ownership & Control (2013): 292

[30] Gallery, Gerry T., Natalie Gallery, and Baljit K. Sidhu. "Quarterly cash flow reporting–is it useful?" InFinsia: journal of Finsia, Financial Services Institute of Australasia 3 (2004): 14-18

[31] Natalya F., et al. "Creating semantic web contents with protege2000." IEEE intelligent systems 16.2 (2001): 60-71

[32] Michael Westergaard, and Fabrizio Maria Maggi. "Declare: A Tool Suite for Declarative Workflow Modelling and Enactment." BPM (Demos) 820 (2011): 1-5.

[33] Tzeremes, V., Gomaa H. : XANA: an end user software product line framework for smart spaces. In: 2016 49th Hawaii International Conference on System Sciences (HICSS). IEEE, (2016).

[34] Pilaka M., Bandara M., Mansoor E. (2019) A Semantic Model Based Framework for Regulatory Reporting Process Management. In: Mehandjiev N., Saadouni B. (eds) Enterprise Applications, Markets and Services in the Finance Industry. FinanceCom 2018. Lecture Notes in Business Information Processing, vol 345. Springer, Cham

AUTHORS

Fethi Rabhi is a Professor in the School of Computer Science and Engineering at the University of New South Wales (UNSW) in Australia. His main research areas are in service-oriented software engineering with a strong focus on business and financial applications. He completed a PhD in Computer Science at the University of Sheffield in 1990 and held several academic appointments in the USA and the UK before joining UNSW in 2000. He is currently actively involved in several research projects in the area of financial software systems and big data analysis.

Manjula Pilaka is a Data Scientist in the School of Computer Science and Engineering at the University of New South Wales(UNSW) in Australia. She completed her MPhil in Computer Science at UNSW and an MBA from Indian Institute of Management(IIM), India. She worked in Industry for 19years in various roles in Australia, USA, UK and India.She is currently actively involved in research areas of Semantic technologies and Machine learning with a strong focus on financial applications.

Madhushi Bandara is working towards her PhD degree at the University of New South Wales, Australia studying the potential of semantic models to capture data analytic related knowledge and assist in analytic solution engineering space. Her research interests are on big data analytics, software architecture and semantic web technologies. She is passionate about teaching software architecture and semantic web technologies and has experience as a lecturer at the University of New South Wales, Australia and University of Moratuwa, Sri Lanka