Pipeline Production Data Model

Jitao Yang
Production System Research Lab, CPPEI
PetroChina Company Limited
Beijing 102206, China
yangjitao@petrochina.com.cn

Yu Fan
Production System Research Lab, CPPEI
PetroChina Company Limited
Beijing 102206, China
fanyugd@petrochina.com.cn

Yinliang Liu
Production System Research Lab, CPPEI
PetroChina Company Limited
Beijing 102206, China
liuyinliang@petrochina.com.cn

Hui Deng
Production System Research Lab, CPPEI
PetroChina Company Limited
Beijing 102206, China
denghui1984@petrochina.com.cn

Yang Lin
Production System Research Lab, CPPEI
PetroChina Company Limited
Beijing 102206, China
lin_yang@petrochina.com.cn

ABSTRACT

With the rapid construction of long-distance oil and gas pipelines, big pipeline network was gradually formed in China and its surrounding areas, therefore the management of pipeline production becomes increasingly complicated and difficult.

In this paper, we propose a data model for pipeline production to support the planning, scheduling, distribution, metering, energy consumption, process technology, professional computing and the other business in pipeline production. We also present a query language that can be used for discovering resources of interest in pipeline production based on the descriptions attached to the resources.

Following our model and based on the actual business needs, we have implemented a pipeline production management system (PPMS). The system provides services for group company, professional companies, regional companies and their affiliated grassroots units, and overseas companies to improve the efficiency and effectiveness of pipeline production management as well as safeguard the accuracy and the completeness of data.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Theory

Keywords

Data Model, Information System, Oil and Gas Pipeline, Pipeline Production Management

1. INTRODUCTION

In recent years, with the rapid construction of long-distance oil and gas pipelines, the big pipeline network was gradually formed in China and its surrounding areas. As of March 2013, a group company owns about 50,000 kilometers long-distance pipelines, of which approximately 33,000 kilometers of natural gas pipelines, 10,000 kilometers crude oil pipelines and 6,700 kilometers refined oil pipelines. The pipelines formed a cross-country and cross-region oil and gas pipeline network. This makes the production management of oil and gas pipeline network become more and more difficult and complex.

Roughly speaking, pipeline production management system is an information system managing the storage and transportation business of long-distance oil and gas pipelines. In particular, PPMS focused on the pipeline production business including: planning management, scheduling and operation, transportation and distribution, professional computing, statistics and analytics etc. In general, a pipeline production management system will be used to:

- standardize the management of pipeline production and operation,
- safeguard the accuracy and the completeness of data,
- improve the efficiency and effectiveness of pipeline production management, as well as
- reduce the burden of grass-roots staffs.

In order to carry out the above mentioned tasks, a pipeline production management system should be able to handle the entities and relationships in pipeline production in an accurate and flexible way, amongst which descriptions and their attachments to resources play an important role.

Combined with years of relevant professional pipeline production business knowledge and the previous experience of building,

operating and managing the information systems for pipeline production, in this paper, we propose a data model for pipeline production defining in a clear and rigorous way all the resources and relationships required to carry out the tasks of a pipeline production management system. The model is established based on the deep understanding of oil and gas pipeline production business and the related technologies, including the planning management, scheduling and operation, transportation and distribution, professional computing, statistics and analytics, Supervisory Control and Data Acquisition (SCADA) system, ERP system and etc. We formalize the model using first order logic, and the model places descriptions as first class citizens. We also present a query language that can be used for discovering resources in pipeline production based on the descriptions of resources.

The rest of the paper is structured as follows: Section 2 introduces the basic concepts of pipeline production, then Section 3 defines the formal data model for pipeline production along with an example, and Section 4 presents the query language. Section 5 discusses the implementation of a pipeline production management system. Section 6 briefly reviews the relevant work. Finally, Section 7 concludes the paper and outlines future work.

2. THE BASIC CONCEPTS OF PIPELINE PRODUCTION

Intuitively, we think of a resource as anything can be identified in pipeline production, such as a pipeline, a station, a place and so on. For the purposes of our discussions, we assume the existence of a set consisting of all resources that one can ever define in pipeline production.

We view each resource is associated with a content and a set of descriptions.

2.1 Content

In this work, by content of a resource r we mean the set of other resources that make up r; each such resource is called a part of r. For example, each pipe segment (seen as a resource in its own right) can be seen as a part of the pipeline. Similarly, each station can be seen as a part of an organization.

2.1.1 Concepts

To understand pipeline production business, basic knowledge of pipeline production is necessary, therefore the definitions of some important concepts in pipeline production are given as follows:

- Station, is the industrial place supporting the operation, maintenance, and monitoring of pipelines. Some of the stations also support the sales of gas or oil and the other pipeline production business. Along the pipelines, the stations provide different functions, for example, there are pump station, compressor station, metering station and so on. The valve chamber, gas storage, and oil depots along the pipelines are also classified into station. Station is the key node along the pipelines.
- Pipe Segment, is the pipe between two stations. Pipe segment is represented by a directed line segment.
- Pipeline, is the pipe connected by a plurality of directed pipe segments.
- Equipment, is the device used in pipeline production, such as Compressor, Chromatographic Analyzer, Flow Meter, Pump, Air Cooler, and other oil and gas pipeline monitoring and control devices.
- Organization, is the oil and gas pipeline production business-related company and the administrative department affiliated to the company.
- Customer, is the buyer of Natural Gas, Crude Oil, Compressed Natural Gas (CNG), or Liquefied Natural Gas (LNG).
- Supplier, is the company of supplying Natural Gas, Crude Oil, Compressed Natural Gas, or Liquefied Natural Gas.
- Transport Location, is the location where the gas or oil was transported away by train, boat, lorry or the other carriers except by the pipelines.

Here, we just list a few import concepts of our understanding in pipeline production, note that, the concepts may be defined differently by the other pipeline production business related organizations.

2.1.2 Relations

In pipeline production, some of the resources are closely related, for instance, stations are connected by pipeline, there are multiple suppliers providing gas or oil for a station and etc. In order to meet the actual business needs, the relations in pipeline production should be described clearly and rigorously. Some of the relations in our pipeline production business are listed as follows:

- Pipe Segment and Station: each pipe segment is defined by the starting station and the ending station.
- Pipeline and Pipe Segment: pipeline is composed by a plurality of pipe segments in a certain order.
- Station and Organization: an organization can manage multiple stations.
- Pipeline and Organization: a pipeline can be managed by multiple organizations.
- Organization, Customer and Supplier: an organization can be a customer and a supplier at the same time.
- Equipment, Station and Organization: an equipment can be associated with multiple stations or multiple pipelines. By default, an equipment belongs to the organization that manages the station where the equipment is located at; if an equipment is associated with multiple stations, then the equipment belongs to the least upper bound organization that can manage all the stations that the equipment is associated with.
- Supplier, Pipeline, Station, and Transportation Location: a supplier could transport its oil/gas to a station (suppose identified by st) by a pipeline; if in the station st, the oil/gas was transported to another place for distribution by train or boat or the other carriers except by pipeline, then the position of the station st is a transport location.
- Turnover between Pipelines: the transported oil/gas could be transferred from one pipeline to another pipeline through a station.

2.2 Description

A description of a resource r is the descriptive information of the resource. For example, the description of a station will include the location and the type of the station, while the description of a pipeline will include the length and the regions that the pipeline goes through and etc. A pipeline, however, can be described from

different points of view, each leading to a different description. As a result, a pipeline might be associated to a set of descriptions (and the same goes for a station). In order to accommodate several descriptions for the same resource, we will treat descriptions as resources in their own right.

2.2.1 Data Points

Data points are the important data that we should take special care of in pipeline production, so that to:

- monitor the operating status of pipelines (*e.g.* through the unit operation parameters);
- meter the transportation or distribution volume of oil/gas (*e.g.* through the flow-meter operation data);
- analyze the gas quality (*e.g.* through the chromatographic analyzer operation data);
- support the professional computing (*e.g.* calculating pipe deposit according to ISO 12213-2:2006 "Natural gas – Calculation of compression factor – Part 2: Calculation using molar-composition analysis" [13]);
- generate different kinds of report forms (*e.g.* Energy Consumption Monthly Report of Natural Gas Station);
- provide decision support by statistics and analytics; and etc.

In pipeline production, to use data points effectively, data points must be associated with descriptions. For example, the rotation speed of compressor is a very important operating parameter used for monitoring the status of compressor; however, if we only have the rotation speed of a compressor (suppose marked by a data item *RotationSpeed*), then the *RotationSpeed* data will be useless, because to know the corresponding compressor, we need to find the identifier of the compressor, the location of the compressor and the other information related to the compressor. Therefore, descriptions should be attached to data points. For the above example, the data point *RotationSpeed* could be associated with *CompressorSN*, *Station*, *Time*, *Date* and the other related properties by descriptions.

2.2.2 Description Reuse

In pipeline production, descriptions should be reusable so that to reduce the workloads of describing similar resources. Generally, we have two ways of reuse:

- reuse without any customizations, which means a description will be reused directly without any modifications. There are thousands and tens of thousands of data acquisition points along the pipelines, some of the data points are the same type, therefore it is not a good idea if we need to create each time a similar description for each data point of the same type. For example, a station could have multiple compressors, the data acquisition points of compressor rotation speed are the same type, these data points could be described by reusing a same description.
- reuse with customizations, which means a description will be reused by extending the description with new properties. For example, to further describe the rotation speed of a compressor *RotationSpeed*, we could attach new properties for instance *InletPressure* and *OutletTemperature* to the description for *RotationSpeed*.

We note that descriptions will be defined according to schemas, for our purposes, we view a schema as consisting of a set of classes and/or a set of properties; additionally, a property could have one or multiple ranges and one or multiple domains. Our definition of classes and properties are similar to the ones in object-oriented modeling, in Description Logics [5], and in semantic web framework [6]. When defining a description, properties may come from one or more schemas.

3. PIPELINE PRODUCTION DEFINITION

3.1 The Language $\mathscr{L}$ of Our Model

The language that we propose for our data model is a function-free first-order language, the predicate symbols are listed as follows:

- $\mathsf{Class}(s,c)$, expresses the fact that c is a class defined in schema s.
- $\mathsf{Property}(s,p)$, expresses the fact that p is a property defined in schema s.
- $\mathsf{Domain}(s,p,c)$, expresses the fact that in schema s class c is the domain of property p.
- $\mathsf{Range}(s,p,c)$, expresses the fact that in schema s class c is the range of property p.
- $\mathsf{IsaCl}(s,c_1,c_2)$, expresses the fact that in schema s class c_1 is a sub-class of class c_2.
- $\mathsf{IsaPr}(s,p_1,p_2)$, expresses the fact that in schema s property p_1 is a sub-property of property p_2.
- $\mathsf{Description}(d,s,p)$, expresses the fact that property p over schema s belongs to description d.
- $\mathsf{PrVal}(i,s,p,j)$ expresses the fact that i has a resource identified by j as value of property p from schema s.
- $\mathsf{ClInst}(i,s,c)$ expresses the fact that i is an instance of class c from schema s.
- $\mathsf{PartOf}(i_1,i_2)$, expresses the fact that i_1 identifies a resource which is part of the resource identified by i_2.
- $\mathsf{DescOf}(d,i)$, expresses the fact that d is a description of i.

We denote the the above predicate symbols defined in first-order logic as $\mathscr{L}$. Before we proceed further with the definition of our data model for pipeline production, let's see how a real example in pipeline production can be represented using $\mathscr{L}$.

3.2 An Example Using $\mathscr{L}$

Consider a SCADA data point "Compressor Rotation Speed", we refer to it by an identifier *crs*. In our formulas, we represent strings by enclosing them between quotes, and resources as the values themselves italicized. Based on these conventions, we have the following formula:

$$\mathsf{ClInst}(crs, s, CompressorRotationSpeed)$$

By the above formula, we understand that *crs* is an instance of class *CompressorRotationSpeed* and *CompressorRotationSpeed* is a class defined in schema *s*:

$$\mathsf{Class}(s, CompressorRotationSpeed)$$

To describe the data point *crs*, we define a description d including multiple properties from schema s_1 and s_2, and description d is represented as follows:

$$\begin{aligned}
&\mathsf{Description}(d, s_1, \mathit{Name})\\
&\mathsf{Description}(d, s_1, \mathit{CompressorSN})\\
&\mathsf{Description}(d, s_1, \mathit{Speed})\\
&\mathsf{Description}(d, s_1, \mathit{Time})\\
&\mathsf{Description}(d, s_1, \mathit{Date})\\
&\mathsf{Description}(d, s_1, \mathit{Station})\\
&\mathsf{Description}(d, s_2, \mathit{InletPressure})\\
&\mathsf{Description}(d, s_2, \mathit{OutletPressure})
\end{aligned}$$

We understand in the above formulas, *Name*, *CompressorSN*, *Speed*, *Time*, *Date*, and *Station* are properties defined in schema s_1, while the properties *InletPressure* and *OutletPressure* are defined by schema s_2. They are represented as:

$$\begin{aligned}
&\mathsf{Property}(s_1, \mathit{Name})\\
&\mathsf{Property}(s_1, \mathit{CompressorSN})\\
&\mathsf{Property}(s_1, \mathit{Speed})\\
&\mathsf{Property}(s_1, \mathit{Time})\\
&\mathsf{Property}(s_1, \mathit{Date})\\
&\mathsf{Property}(s_1, \mathit{Station})\\
&\mathsf{Property}(s_2, \mathit{InletPressure})\\
&\mathsf{Property}(s_2, \mathit{OutletPressure})
\end{aligned}$$

We now could assert that d is a description of *crs* by a formula:

$$\mathsf{DescOf}(d, \mathit{crs})$$

Finally, we could attach values to the properties of *crs* as follows:

$$\begin{aligned}
&\mathsf{PrVal}(\mathit{crs}, s_1, \mathit{Name}, \text{“Compressor Rotation Speed”})\\
&\mathsf{PrVal}(\mathit{crs}, s_1, \mathit{CompressorSN}, \text{“UZS-001”})\\
&\mathsf{PrVal}(\mathit{crs}, s_1, \mathit{Speed}, \text{“8 000”})\\
&\mathsf{PrVal}(\mathit{crs}, s_1, \mathit{Time}, \text{“13:12:11”})\\
&\mathsf{PrVal}(\mathit{crs}, s_1, \mathit{Date}, \text{“2013-11-01”})\\
&\mathsf{PrVal}(\mathit{crs}, s_1, \mathit{Station}, \text{“UZS”})\\
&\mathsf{PrVal}(\mathit{crs}, s_2, \mathit{InletPressure}, \text{“5 000 000”})\\
&\mathsf{PrVal}(\mathit{crs}, s_2, \mathit{OutletPressure}, \text{“8 000 000”})
\end{aligned}$$

Note that, the above description d could be reused to describe other SCADA data points. For example, a compressor rotation speed crs_2 of another compressor, which is the same type of *crs*, can be described by d as follows:

$$\mathsf{DescOf}(d, \mathit{crs}_2)$$

We then can attach different values to the properties of crs_2 using PrVal as follows:

$$\begin{aligned}
&\mathsf{PrVal}(\mathit{crs}_2, s_1, \mathit{Name}, \text{“Compressor Rotation Speed”})\\
&\mathsf{PrVal}(\mathit{crs}_2, s_1, \mathit{CompressorSN}, \text{“UZS-002”})\\
&\mathsf{PrVal}(\mathit{crs}_2, s_1, \mathit{Speed}, \text{“9 000”})\\
&\mathsf{PrVal}(\mathit{crs}_2, s_1, \mathit{Time}, \text{“23:12:11”})\\
&\mathsf{PrVal}(\mathit{crs}_2, s_1, \mathit{Date}, \text{“2013-12-01”})\\
&\mathsf{PrVal}(\mathit{crs}_2, s_1, \mathit{Station}, \text{“UZS”})\\
&\mathsf{PrVal}(\mathit{crs}_2, s_2, \mathit{InletPressure}, \text{“4 500 000”})\\
&\mathsf{PrVal}(\mathit{crs}_2, s_2, \mathit{OutletPressure}, \text{“7 000 000”})
\end{aligned}$$

3.3 The Axioms $\mathscr{A}$ of Our Model

We now present the axioms of our model. The axioms of our model will be used to capture the meaning of the predicate symbols, as well as capture the implicit knowledge in pipeline production. Variables in the axioms are universally quantified.

(A1) If property p has class c as its domains in a schema s, then p and c must be defined in s:

$$\mathsf{Domain}(s, p, c) \rightarrow (\mathsf{Property}(s, p) \wedge \mathsf{Class}(s, c))$$

(A2) If a property p has class c as its ranges in a schema s, then p and c must be defined in s:

$$\mathsf{Range}(s, p, c) \rightarrow (\mathsf{Property}(s, p) \wedge \mathsf{Class}(s, c))$$

(A3) If c_1 is a sub-class of c_2 in a schema s, then c_1 and c_2 must be defined in s:

$$\mathsf{IsaCl}(s, c_1, c_2) \rightarrow (\mathsf{Class}(s, c_1) \wedge \mathsf{Class}(s, c_2))$$

(A4) If p_1 is a sub-property of p_2 in a schema s, then p_1 and p_2 must be defined in s:

$$\mathsf{IsaPr}(s, p_1, p_2,) \rightarrow (\mathsf{Property}(s, p_1) \wedge \mathsf{Property}(s, p_2))$$

(A5) If i is an instance of class c over schema s, then c must be defined in schema s :

$$\mathsf{ClInst}(i, s, c) \rightarrow \mathsf{Class}(s, c)$$

(A6) If a description d contains a property p over schema s, then p must be defined in schema s :

$$\mathsf{Description}(d, s, p) \rightarrow \mathsf{Property}(s, p)$$

(A7) Sub-class is reflexive:

$$\mathsf{Class}(s, c) \rightarrow \mathsf{IsaCl}(s, c, c)$$

(A8) Sub-class is transitive:

$$(\mathsf{IsaCl}(s, c_1, c_2) \wedge \mathsf{IsaCl}(s, c_2, c_3)) \rightarrow \mathsf{IsaCl}(s, c_1, c_3)$$

(A9) Sub-property is reflexive:

$$\mathsf{Property}(s, p) \rightarrow \mathsf{IsaPr}(s, p, p)$$

(A10) Sub-property is transitive:

$$(\mathsf{IsaPr}(s, p_1, p_2) \wedge \mathsf{IsaPr}(s, p_2, p_3)) \rightarrow \mathsf{IsaPr}(s, p_1, p_3)$$

(A11) If i is an instance of class c_1 from schema s, and c_1 is a sub-class of c_2 in s, then i is also an instance of class c_2 from schema s :

$$(\mathsf{ClInst}(i, s, c_1) \wedge \mathsf{IsaCl}(s, c_1, c_2)) \rightarrow \mathsf{ClInst}(i, s, c_2)$$

(A12) If p_1 is a sub-property of p_2 in s, and j is a p_1-value of i, then j is also a p_2-value of i:

$$\begin{aligned}
&(\mathsf{IsaPr}(s, p_1, p_2) \wedge \mathsf{PrVal}(i, s, p_1, j))\\
&\rightarrow \mathsf{PrVal}(i, s, p_2, j)
\end{aligned}$$

(A13) If d describes i that is part of j, then d describes j too:

$$(\mathsf{DescOf}(d, i) \wedge \mathsf{PartOf}(i, j)) \rightarrow \mathsf{DescOf}(d, j)$$

This axiom transfers descriptions from parts to the whole.

We denote the above set of axioms as $\mathscr{A}$, and we denote the theory defined by $\mathscr{L}$ and $\mathscr{A}$ as $\mathscr{T}$.

In pipeline production, an interpretation I is created by the contents and the descriptions manually inserted by users when they record information about pipeline production or gathered from the relevant systems such as SCADA system, ERP system etc. The resulting pipeline production is then given by applying the axioms to these facts. In order to make this concept more precise, we re-write the axioms $\mathscr{A}$ of our data model for pipeline production in the form of a positive datalog program $D_{\mathscr{A}}$ as follows:

$$
\begin{array}{lcl}
\mathsf{Property}(s,p) & :- & \mathsf{Domain}(s,p,c) \\
\mathsf{Class}(s,c) & :- & \mathsf{Domain}(s,p,c) \\
\mathsf{Property}(s,p) & :- & \mathsf{Range}(s,p,c) \\
\mathsf{Class}(s,c) & :- & \mathsf{Range}(s,p,c) \\
\mathsf{Class}(s,c_1) & :- & \mathsf{IsaCl}(s,c_1,c_2) \\
\mathsf{Class}(s,c_2) & :- & \mathsf{IsaCl}(s,c_1,c_2) \\
\mathsf{Property}(s,p_1) & :- & \mathsf{IsaPr}(s,p_1,p_2) \\
\mathsf{Property}(s,p_2) & :- & \mathsf{IsaPr}(s,p_1,p_2) \\
\mathsf{Class}(s,c) & :- & \mathsf{ClInst}(i,s,c) \\
\mathsf{Property}(s,p) & :- & \mathsf{Description}(d,s,p) \\
\mathsf{IsaCl}(s,c,c) & :- & \mathsf{Class}(s,c) \\
\mathsf{IsaCl}(s,c_1,c_3) & :- & \mathsf{IsaCl}(s,c_1,c_2),\ \mathsf{IsaCl}(s,c_2,c_3) \\
\mathsf{IsaPr}(s,p,p) & :- & \mathsf{Property}(s,p) \\
\mathsf{IsaPr}(s,p_1,p_3) & :- & \mathsf{IsaPr}(s,p_1,p_2),\ \mathsf{IsaPr}(s,p_2,p_3) \\
\mathsf{ClInst}(i,s,c_2) & :- & \mathsf{ClInst}(i,s,c_1),\ \mathsf{IsaCl}(s,c_1,c_2) \\
\mathsf{PrVal}(i,s,p_2,j) & :- & \mathsf{IsaPr}(s,p_1,p_2),\ \mathsf{PrVal}(i,s,p_1,j) \\
\mathsf{DescOf}(d,j) & :- & \mathsf{DescOf}(d,i),\ \mathsf{PartOf}(i,j)
\end{array}
$$

Given an interpretation I which can be seen as a set of facts of pipeline production, then the above rules in $D_{\mathscr{A}}$ will be applied in order to derive the minimal model of $D_{\mathscr{A}}$ containing I. The minimal model will be a larger set of facts containing I as well as all the consequences of I according to $D_{\mathscr{A}}$. Based on the logical programming [7], the minimal model exists and is unique.

DEFINITION 1. ***(Pipeline Production)*** *Let I be any interpretation of $\mathscr{L}$, we call pipeline production over I, denoted* PP*, the minimal model $\mathscr{M}(D_{\mathscr{A}},I)$ of $\mathscr{A}$ that contains I.*

4. QUERY THE DATA OF PIPELINE PRODUCTION

When searching the data of pipeline production, an intuitive and straightforward way of expressing the user's information need is to relate description to the sought resource.

For example, to search all the compressors in the station "UZS", the user could use a query like:

$$(\exists\ ?csn)\quad \mathsf{PrVal}(?csn, s, LocatedIn, \text{“UZS”})$$

In the above query, $?csn$ is a variable representing the compressor serial number, *LocatedIn* is a property defined in schema s.

For allowing queries to state simple conditions on property values, we consider any completion of PP endowed with six built-in relations, namely the $=$, $\neq$, $>$, $<$, $\leq$ and $\geq$ relations; we consider these relations as built-in predicates, therefore not subject to the completion of PP.

To exemplify, let us consider again the "Compressor Rotation Speed" example. When searching for the time when the compressor "UZS-001" was not running in the day of 2008-09-25, the user could use a query Q_t like:

$$
\begin{array}{ll}
(\exists\ ?t) & \mathsf{PrVal}(?crs, s, Time, ?t) \wedge \\
& (\mathsf{PrVal}(?crs, s, Speed, ?rs) \wedge (?rs < \text{“100”})) \wedge \\
& \mathsf{PrVal}(?crs, s, CompressorSN, \text{“UZS-001”}) \wedge \\
& \mathsf{PrVal}(?crs, s, Date, \text{“2008-09-25”})
\end{array}
$$

In the above query, $?t$, $?crs$ and $?rs$ are variables occur in Q_t, and $?t$ represents the time, $?crs$ represents the identifier of compressor rotation speed, $?rs$ represents the compressor rotation speed. Note that, the compressor rotation speed can be used to judge the running status of a compressor, *i.e.* if the rotation speed was less than 100, we consider the compressor was not running.

DEFINITION 2. ***(Query over Pipeline Production)*** *A* query over *pipeline production* PP *is well-formed formula $Q(x_1,\ldots,x_n)$ of $\mathscr{L}$, in which $x_1,\ldots,x_n$ are free variables and $n \geq 1$.*

The answer of a query with n ($n \geq 1$) free variables is the set of resources $\langle r_1,\ldots,r_n\rangle$ such that, when every variable x_i is bound to the corresponding resource r_i, the resulting formula of $\mathscr{L}(r_1,\ldots,r_n)$ is true in PP. Formally, we have the following definition for the answer of a query.

DEFINITION 3. ***(Answer of A Query)*** *The* answer *of a query $Q(x_1,\ldots,x_n)$ over pipeline production* PP *is given by:*

$$answer(Q,\mathsf{PP}) = \{\langle r_1,\ldots,r_n\rangle \mid Q(r_1,\ldots,r_n) \in \mathsf{PP}\}$$

5. PIPELINE PRODUCTION SYSTEM

We have adopted first-order logic for modeling pipeline production. The choice of logic was deliberate in order to be able to describe the pipeline production without being constrained by any technical considerations. However, the goal of our work is to contribute to the *management system* of pipeline production. Therefore, we now consider how our model can be implemented. We consider two different scenarios:

- The first scenario consists in implementing the model with relational database and computing the completion of pipeline production via a datalog engine [8, 9, 10, 11, 12]. This implementation is conceptually straightforward, as an interpretation of $\mathscr{L}$ consists just of a set of relations that can be implemented as tables.
- The second scenario consists in implementing the model with Resource Description Framework (RDF) [1], and using an RDF inference engine for computing the completion of the pipeline production. This implementation is conceptually much more complicated, as it requires translating the relations and the axioms of the model in RDF.

The first scenario exploits the well-established relational technology, including the optimized query processing of SQL. This guarantees scalability and robustness of the implementation. The second scenario benefits from the fact that RDF is a generally accepted representation language in the context of the Semantic Web. Although RDF has not yet achieved the maturity of relational technology, tools for managing RDF graphs have been intensely researched and developed in the last decade, and are now reaching a significant level of technological maturity. Such tools include systems for the persistence of large RDF graphs, for instance [3], RDF inference engines and optimized query processing engines for SPARQL [2], *e.g.* [4].

5.1 Relational Implementation

By choosing an implementation strategy based on relational technologies, we can benefit at a minimal effort of the scalability and the optimized query evaluation of relational database management systems (RDBMSs).

A simple strategy for implementing the model could then consist of the following steps:

- Store the initial interpretation I of pipeline production to a relational database $RDB(I)$; the mapping from I to $RDB(I)$ is straightforward.
- Compute the completion of $RDB(I)$ to obtain the database $RDB(\mathscr{M}(D_{\mathscr{A}},I))$ via a datalog engine [8, 9, 10, 11, 12]; this requires adding tuples to the tables in $RDB(I)$ using the inference mechanism that we have described in $\mathscr{A}$.
- Map each query Q against the pipeline production to an equivalent SQL query $SQL(Q)$.

To implement our data model for pipeline production by relational database, we should pay attention to the design of algorithms for maintaining $RDB(\mathscr{M}(D_{\mathscr{A}},I))$ in the condition of user updates. We also should take care in choosing a suitable datalog engine.

5.2 RDF based Implementation

RDF is a knowledge representation language for describing resources using triples of the form (subject, predicate, object). In a triple, the subject can be a URI or a local identifier (also called blank node) for unnamed entities; the predicate can only be a URI; the object can be a URI, a local identifier or a literal (*i.e.* a string of characters). The predicate in an RDF triple specifies how the subject and the object of the triple are related.

A set of RDF triples is called an RDF graph. This graph is obtained by interpreting each triple as a labelled arrow having the subject as its source, the object as its target and the predicate as its label. RDF has a formal semantics on top of which an inference mechanism is defined. This mechanism allows expanding a given RDF graph by adding to it the new triples that can be inferred from existing ones.

In order to implement our model in RDF, we must provide means for:

- mapping the relations of $\mathscr{L}$ into equivalent RDF graphs,
- mapping the inference mechanism of our data model into that of RDF, and
- translating the query language of our model to SPARQL.

5.3 System Development

As we have already observed, implementing our model using RDF is more complicated, while implementing our model using relational technologies is straightforward. For this reason and due to the time limit for the project of pipeline production management system, we implemented the pipeline production data model using relational technologies.

Based on the investigation report and solution design, pipeline production management system should provide the following business functions for the users:

- Planning Management, manages yearly agreement intake and transportation volume of gas/oil, yearly planned production volume of gas/oil, subdivided monthly production plan, and etc.

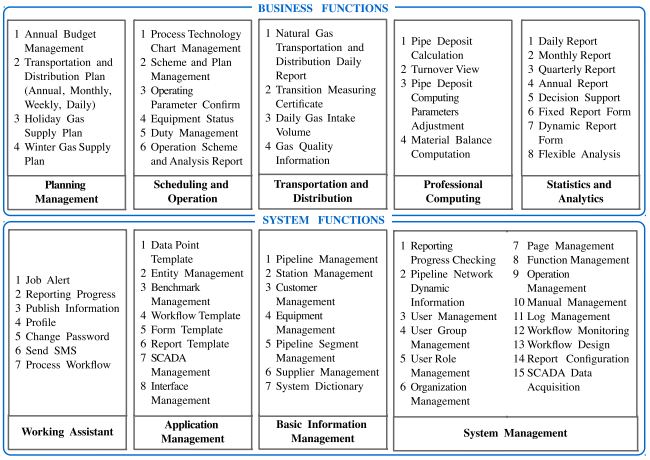

Figure 1: Function Architecture of Pipeline Production Management System

- Scheduling and Operation, manages station daily pipeline production operating data, production logs, unit operating data, flow-meter operating data, chromatographic analyzer operating data, as well as generates production operating daily report and hourly parameter summary table.
- Transportation and Distribution, manages the daily gas/oil intake volume, daily gas/oil distribution volume, gas/oil quality information and etc.
- Professional Computing, provides pipeline production professional computing services. For instance, pipe deposit calculation according to the ISO 12213-2:2006 "Natural gas – Calculation of compression factor – Part 2: Calculation using molar-composition analysis" [13].
- Statistics and Analytics, generates different kinds of reports including production daily report, production monthly report, operation daily report, energy consumption report, and so on for the group company headquarters, the regional companies, and the vendors, as well as provides decision support.

The pipeline production business functions are supported by the following system functions:

- System Management, configures the system functions, such as the user group creation based on users' authority layers, user role configuration based on user's duties, work-flow management based on business process, and etc.
- Basic Information Management, manages the basic information of the system such as pipeline basic information maintenance, pipeline network and station visualization, pipeline running dynamic visualization, supplier and customer information management, and etc.
- Application Management, manages the configurable applications such as the configuration of different form templates for different stations, configuration of different reports based on different companies' requirement, and etc.
- Working Assistant, enhances production management efficiency by the functions of job alerts, work-flow process, system information publish, short message service, personal information management, and the other auxiliary functions.

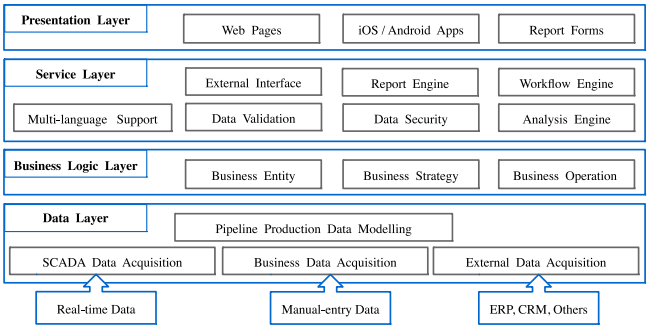

Figure 2: System Architecture of Pipeline Production Management System

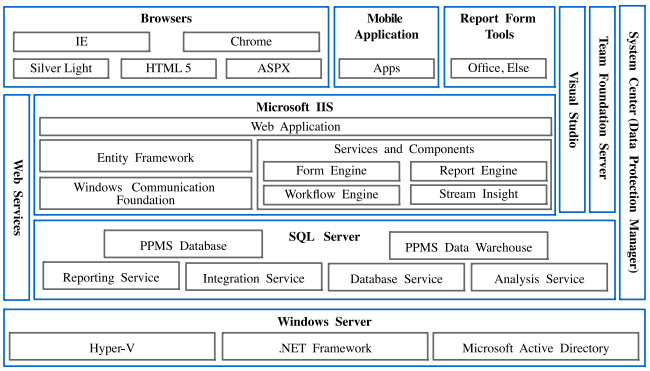

Figure 3: Software Architecture of Pipeline Production Management System

The function architecture of the pipeline production management system is summarized in Figure 1.

The data of PPMS comes from SCADA system, manual entry, and the external systems (ERP, CRM and etc) of the enterprise, the system architecture of pipeline production management system is presented in Figure 2.

PPMS was developed with .NET and the related softwares and technologies, the software architecture of pipeline production management system is given by Figure 3.

The hardwares for supporting the running of PPMS are listed in Table I.

A simplified ER diagram of PPMS is demonstrated in Figure 4.

A few ways were used to ensure the security of pipeline production management system:

- communications security: strict access control was provided by a firewall that isolates the enterprise internal network from the outside world; Demilitarized Zone (DMZ) and HTTPS encryption were used to protected external access to PPMS; in addition, hardware encryption (USB key) was used for PPMS user authentication.
- data security: local and off-site data backups were performed regularly (in seconds).
- system application security: hot standby mechanism was established for important services; virtualization technology was used to enhance the high availability of web servers.

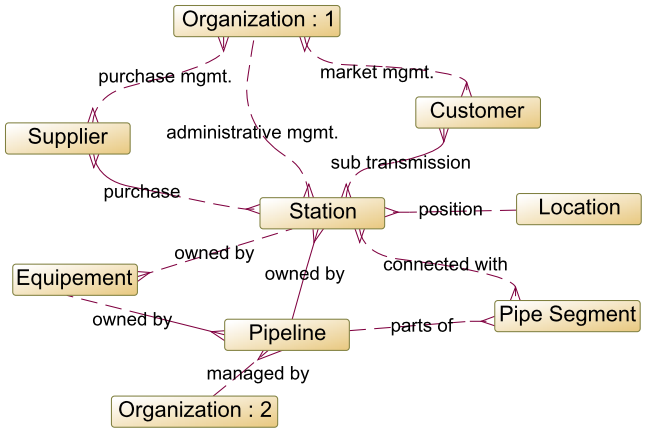

Figure 4: A Simplified ER Diagram of Pipeline Production Management System

Table I: The Hardwares for Running PPMS

Device	Device Requirements and Configuration	Qty
Web Server	10 core 2.26GHz Intel Xeon E7 series processor × 4, 256GB DDR3 memory, 8GB HBA card × 2, 600GB SAS hot plug hard drive × 5, Hyper-V supported	6
Report Form Server	10 core 2.26GHz Intel Xeon E7 series processor × 4, 64GB DDR3 memory, 600GB SAS hot plug hard drive × 5, Hyper-V supported, dual backup required	2
Operation Monitoring Server	ditto, no dual backup required	2
Application Server	ditto, dual backup required	3
Data Acquisition Server	ditto	2
Front End Server	8 core 2GHz Intel Xeon E5 series processor × 2, 32GB DDR3 memory, 600GB SAS hot plug hard drive × 5	1
Database Server	10 core 2.26GHz Intel Xeon E7 series processor × 4, 256GB DDR3 memory, 8GB HBA card × 2, 600GB SAS hot plug hard drive × 5, Hyper-V supported, double cluster deployment	4
ESI Load Forecasting Server	10 core 2.26GHz Intel Xeon E7 series processor × 4, 64GB DDR3 memory, 600GB SAS hot plug hard drive × 5, Hyper-V supported	1
Backup Server	8 core 2GHz Intel Xeon E5 series processor × 2, 32GB DDR3 memory, 600GB SAS hot plug hard drive × 5	1
Storage Device	Dual controller, buffer capacity of 24G, 16 front-end 8Gb fiber ports, 8 back-end 6Gb ports, 4 Gigabit NAS network ports; SAN architecture drive bare capacity 30TB, NAS architecture drive bare capacity 20TB; 24-port optical switch, configure and activate 16 ports	1
Load Balancer	12 Gigabit ports, 4 core processor, 4Gbps throughput, the maximum number of concurrent connections: 8000000, 8GB DDR3 memory, double storage medium	2
Ethernet Switch	48-port Gigabit switch, redundant power supply	2
KVM	16 ports, with remote management function	1
Disk Array	Logical backup capacity more than 100TB, 2 BGb FC ports, 2 Gigabit ethernet ports	1

The pipeline production management system has been launched for a few months, providing around 276 professional services for managing the group company's pipeline production business within China. The number of ingested data is around 622800 items per day. The system was also extended to manage the pipeline production of Turkmenistan, Uzbekistan, Kazakhstan and China gas pipelines and the pipeline production of Sino-Burma gas pipelines.

6. RELATED WORKS

To the best of our knowledge, our model is the first data model specializing in the production management of pipelines. Although there are some models for pipelines such as Pipeline Open Data Standard (PODS) [14], ArcGIS Pipeline Data Model (APDM) [16], and PODS ESRI Spatial [15], these models focus on the management of pipeline integrity, *i.e.* integrated with Geographic Information System (GIS) information, these models help pipeline operators to collect, verify, analyze, and maintain the information

about physical segment of pipeline, so that to support the re-route, change of service, abandonment, removal, repair, and replacement of pipeline [14].

Entity-Relationship (ER) model [17] is a data model for describing data in an abstract and conceptual way, the essential elements of ER model are: Entity Set, Relationship Set, and Attribute. Entity is an object that can be uniquely and distinctly identified, Entity Set is a set of entities of the same type that has the common properties. Relationship is the association among entities, Relationship Set is a mathematical relationship among $n \geq 2$ entities. Attribute is the descriptive information about an entity or a relationship. The ER model describes only the static view of data. Unified Modeling Language (UML) [18] is a widely accepted object-oriented modeling language that has many components to model different aspects of an entire system graphically; Class Diagram is the closest component of UML corresponds to ER Diagram, but several differences [19]. In our model, we use relations to express the basic facts in pipeline production, and rely on a first-order theory to derive information implicit in the given facts, under certain axioms; descriptions are placed as first class citizens and are defined independently of the resources they might be associated with, this is different from the systems often used to implement ER models, which generally assume that attributes of an object are stored in the object; in addition, our model supports the reuse of descriptions. The choice of logic for modeling is motivated by the desire of generality, which includes freedom from any technological constraint.

Enterprise Resource Planning (ERP) is a system focused on business management, to improve the efficiency and effectiveness of business processes so that to reach the business goals. Customer Relationship Management (CRM) is a system for a company to manage current and future customer interactions, to improve customer relations so that to increase brand loyalty and profits. However, neither ERP nor CRM can satisfy the requirements of our pipeline production management. In fact, part of the customer information from CRM and some of the ERP modules such as PM (Plant Maintenance), MM (Materials Management) and PS (Project Systems) are the data sources of pipeline production management system, in turn, PPMS supports the FI (Financial Accounting) module of ERP.

7. CONCLUSIONS

In this paper, we have defined a data model for pipeline production based on two basic concepts: content and description. The two concepts are expressed by a certain set of predicates using a first-order logical approach, the axioms of the model are defined to fix the semantics of the predicates and to capture the implicit knowledge in pipeline production. In our model, descriptions are placed as first class citizens with their own identifiers and are defined independently of the resources they might be attached to, additionally descriptions are flexible for extension and reuse. We also present a query language for discovering resources of interest in pipeline production based on the descriptions attached to the resources.

Our model provides a convenient and flexible way for describing the concepts, the properties, and the relations in pipeline production. The model also leads to an efficient communication between the business people defining the pipeline production management requirements for an information system and the technical people developing the information system in response to those requirements.

Following our model, we implemented a pipeline production management system for managing the group company's pipeline production business in China as well in some overseas regions. The pipeline production management system received very high appraisal from the users of the group company, the regional companies, and the overseas companies.

Regarding further work, we will continue improving our model. Furthermore, there is one direction which is promising in our opinion that the group company has many different information systems, which are categorized into ERP Systems, Petroleum Exploration and Development Systems, Refining and Marketing Systems, Service and Support Systems, Infrastructure and Security Systems, and so on. We are currently working towards the extension of our model to manage the big data across these above mentioned systems to allow all information systems of the enterprise to operate together in a cooperative manner, so that to maximize the overall data management and analysis benefit to the enterprise.

8. REFERENCES

[1] Resource Description Framework (RDF). http://www.w3.org/RDF/

[2] SPARQL 1.1 Overview, W3C Recommendation 21 March 2013. http://www.w3.org/TR/sparql11-overview/

[3] Semantic Web Development Tools. http://www.w3.org/2001/sw/wiki/Tools

[4] SPARQL Implementations. http://esw.w3.org/SparqlImplementations

[5] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi and Peter F. Patel-Schneider. The description logic handbook: theory, implementation, and applications. Cambridge University Press, 2003.

[6] Frank Manola and Eric Miller (Editors). RDF Primer, W3C Recommendation 10 February 2004. http://www.w3.org/TR/rdf-primer/

[7] J. W. Lloyd. Foundations of Logic Programming. Springer-Verlag New York, Inc., 1987.

[8] Datalog. http://www.ccs.neu.edu/home/ramsdell/tools/datalog/

[9] Clojure Datalog. https://code.google.com/p/clojure-contrib/wiki/DatalogOverview

[10] IRIS Reasoner. http://iris-reasoner.org/

[11] 4QL. http://4ql.org/

[12] Datalog Educational System (DES). http://www.fdi.ucm.es/profesor/fernan/DES/

[13] ISO 12213-2:2006 Natural gas – Calculation of compression factor – Part 2: Calculation using molar-composition analysis. http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=44411

[14] Pipeline Open Data Standard. http://www.pods.org/

[15] PODS ESRI Spatial. http://www.pods.org/75/PODS%20ESRI%20Spatial%20Geodatabase/

[16] ArcGIS Pipeline Data Model (APDM). http://www.esri.com/industries/pipeline/community/datamodel

[17] Peter Pin-shan Chen. The Entity-Relationship Model: Toward a Unified View of Data. ACM Transactions on Database Systems, volume 1, pages 9–36, 1976.

[18] Unified Modeling Language (UML). http://www.omg.org/spec/UML/

[19] Bernadette Marie Byrne and Yasser Shahzad Qureshi. The Use of UML Class Diagrams to Teach Database Modelling and Database Design. Proceedings of the 11th International Workshop on the Teaching, Learning and Assessment of Databases, University of Sunderland, 5 July 2013.