Extending RDF for Decision Making

Wafik Farag
Wafik@SkyPrise.com

Position

Decision making requires sound information to target successful results. Generation of knowledge not data is what is missing. Similarly, knowledge freshness is key is making correct timely decisions. Whether the aim is better productivity or target identification and validation, the end goal is making sound decisions leading to successful results. Data-overloaded is blocking the view to a result dashboard. A number of initiatives still focus on data because of unsolved problems like – data formats, data models, data integration, data access, data storage, etc., though most agree that the goal is to turn data into knowledge. Our position is an end-user must have access to a dynamic information platform that would enable them to manage and share knowledge. I introduce DiBase (Dynamic Information/Integration-Base) a dynamic information integration and collaboration platform that empowers users to analyze reaching better assessments. One key feature in DiBase is that is allows new data, analysis tools, or results that are not predefined to be integrated on-the-fly. This enables new knowledge to be available to use and build upon in a timely fashion and is what we term "Knowledge freshness". Knowledge freshness enhances collaboration – active collaboration where other users can add, test, and build upon existing knowledge. Active collaboration leads to improved decision making process. The same dynamic integrity engine that operates on data schema will be extended to the RDF construct to ensure the integrity of the data stores and the ability for end-users to query data out of RDF.

Integration Approaches – 2 schools of thought

For sound decision making, one requires to have all available data and analysis tools to conduct what-if analysis generating meaningful results. Part of the focus on integration issues is that it plays a significant role bringing desperate data sources integrated as a single view for analysis. Weather this data is from a raw instrument output, data resulting as an output from a computation, or data from another data source. In most cases the new data is not predefined, hence the integration challenges. Two schools of thought address approaches on how to integrate data.

Compatibility and Suppleness – Dynamic

Ability for systems to share data and information and extend to new knowledge not defined a priori. This approach embraces the dynamic school of thought that change is inevitable and a system must be able to extend and incorporate new data preserving system freshness. One can't design or standardized what one has not thought of or discovered yet, hence the dynamic approach works well in evolving environments. RDF addresses the need to extend a set of constructs to accommodate new data constructs. However the challenge is how to ensure the integrity of the newly added data constructs to be in sync with existing data constructs. This resembles the same issues surrounding

XML-databases. XML databases lack an integrity engine for ensuring system correctness. XML is a markup language and not a data model. Existing data models such as relational or object oriented ensure correctness but both are static. Static means no new undefined attributes/relations or objects can be added without a re-design.

Compliance and Standards – Static

Ability for systems to share only pre-specified data and information according to a system standard/design/formats/etc. No new undefined knowledge is allowed. This approach embraces setting standards and enforcing those standards ensuring data conformity across systems.

For systems to operate in evolving environments, knowledge freshness becomes key in the decision making process. Not only data integration needs attention, but integration of analysis tools, and results obtained – that is, information integration. Information integration leads to knowledge freshness, because all information elements are stored and tracked inside the system to be re-used and built upon. Consequently, knowledge generation is achieved by using existing knowledge as building blocks to obtain and store new, not previously defined, results. This is in contrast to storing of data in predefined slots in a database. Hence the first approach of dynamic information integration becomes critical in evolving domains to achieve sound decisions.

DiBase: A Dynamic Information Integration and Collaboration Platform:

DiBase is a platform that integrates several models operating cohesively under one platform for managing dynamic knowledge. A model exists for each information element such as data, functions, or results. Thus, DiBase includes: a data-model, a function-model, and a result-model and an overarching model for the interaction and integration of the different information elements all under one platform. A model is an imitation or emulation of a real life scenario. For example, a data model mimics how data elements and relationships are mimicking relationships among people or objects in real life. All models in DiBase are dynamic to mirror changes that occur in the domains understudy. With the dynamic approach of modeling in DiBase, knowledge freshness preserves a true aspect for modeling evolving domains such as proteomics in life sciences.

Information Silo

In evolving domains new analysis tool, data, or innovations needs to be quickly integrated for timely decisions. Whether it is data, applications, functions, workflows, or results one needs to have all those information elements available in one system to easily integrate together. It is when analysis tools operate on data that knowledge is generated. DiBase makes “What you have work better together”. This is critical in dynamic domains where information elements change and need to be tracked.

Existing software systems assume a single data source used by one or a set of fixed analysis tools to produce a result. No transparency as to which version of the analysis tools are used. At the same time, results are usually not stored, and if they are stored only

the number or image is stored as a data item. In many cases, the result is as important as to how it was obtained. Thus, for sound decision making results need to be stored together with how they were obtained for "what-if" analysis such as what sequence of functions where used, with which input parameters to each function, and the data sets used by each function in the workflow. This produces stored results that are more meaningful for re-use and investigation and not to be believed at face value. The assumptions enforced in obtaining a result are in many times as important as the result itself. This requires a system where all information elements are tracked and available via one system. Information silo is eliminated when a system tracks which analysis tools (i.e. applications, functions, or workflow of the mix), used which data sets to produce which results. DiBase manages the information spectrum via a web-based system to store, access, search, execute, share, or re-run different information elements.

DiBase is a web-based software platform that enables end-users to:

1. **Integrate** existing proprietary databases, public databases, unstructured data sources, new data from results, application, functions, workflows, or results to be accessible through a single web-based system.
2. **Analyze** complex computations, create and share workflows, re-run previously stored result for "what-if" analysis.
3. **Track** analysis activity, data, tools, workflows, and results for compliance to federal mandated standards (e.g. HIPPA, 21, CFR 11) or for intellectual property requirements.
4. **Collaborate** and selectively securely share results, custom functions, process workflows, stored procedures, or data with other users.

Extending DiBase to include RDF

Existing version of DiBase does not accommodate RDF constructs as one of the data types used in its data model like XML, relational, .csv, etc. By extending DiBase to include RDF brings several benefits to the RDF community including:

1. Ensuring the integrity of the RDF within a data model needed in a dynamic collaborative setting.
2. Enables end-users without a priori knowledge about the system to search and access data out of RDF-like constructs.
3. Extending RDF constructs to be accessible to other data types for integration.
4. Utilizing functionality used for other data sources to be quickly extended to data stored in RDF-like constructs.
5. Results obtained from RDF data sets can be tracked and re-run reaching a new level of decision making.

Conclusion:

For systems to support domains where changes occur on continuous bases, the typical re-design cycle experienced by existing software systems becomes obsolete. In a dynamic domain, the decision making process depends on the freshness of the knowledge inside the system. Hence, a system needs to be dynamic and a model needs to exist for each information element managed by the system such as data, functions, or results. Once a

system is able to manage influx of information elements, more accurate assessments are achieved. At the same time, these assessments need to be tracked to be compared and re-investigated at later time, which necessitates a result model to co-exist with other models eliminating the information silo effect. Thus more sound decision are achieved, shared, and verified over time. DiBase product currently interfaces with several data models and extending it to RDF brings the dynamic information integration and collaboration features to the community enhancing the decision making process.