Q&A FROM THE WEBINAR: CONNECTING THE DOTS WITH DYNAMICALLY LINKED DATA FROM DIVERSE SOURCES

Note: In the following TBI is used for TopBraid Insight (TBI)

Q1: What does it take to add (or remove) a datasource? In other words, how easy is it to create a new set of data sources to explore, or change one?

A1: All you need to do is to add a new data source is to connect its schema to the common ‘concept model’ used by your ConnectSet. If the data source brings brand new concepts or attributes that are not yet described by the concept model, extending the model is easy. You simply define new classes and properties. There is no need to change how existing data sources are connected or to make any other modifications.

Recall that in TopBraid Insight, we use the term ConnectSet to define the integration of a specific set of data sources. Let’s say that, like in our example, you have a ConnectSet with four data sources. When bringing a new data source, you may simply add it to the existing ConnectSet or you may want to create a brand new ConnectSet with four existing sources plus the new one or with some subset of existing sources plus the new one. The choice is yours and it depends on the needs of your user community. There will also be cases where you may want to have different ConnectSets for exactly the same data sources because you may want to expose different parts of the underlying data or use a different concept model. TBI gives you this flexibility. To remove a data source you simply remove the statement that says that it is included in a ConnectSet.

Q2: You have shown data integration working with some limited Life Sciences data. Does the tool or product work for other domains? And, if so what is needed and how much effort is involved?

A2: TBI is a generally applicable product that can be used in any industry or domain with respective data sources of interest. What makes it industry specific is the model used and, of course, the data sources. In our example, we used drugs. As mentioned in the presentation, it could have been equipment, product data or equipment failures. Or it could be financial information, and so on.

So, using it with a different industry involves creating ConnectSets for some data sources in your specific domain. The integration work to set up a ConnectSet can be done by domain experts without requiring IT support.

Once this is done, as we showed in the demo, a new ExploreSpace is created with just a click. And you be ready to use TBI in another domain.

Q3: ConnectSet is Global Metadata and ExploreSpace is a DataMart - translated to the familiar concepts?

A3: Yes, an ExploreSpace can be thought of as a dynamically created datamart. A ConnectSet contains metadata necessary to create ExploreSpaces. This includes a unifying model of the concepts of interest within the diverse data sources. Different ConnectSets can use different common (unifying) or canonical models.

Q4: Data Virtualization sparks performance impact paranoia in IT people. Please share some measurements in that area.

A4: TBI optimizes queries sent to each data source. It also has planning capabilities to help decide when to send queries to sources in parallel or when to query one data source first and then query another one second after gaining information from the prior data source that will allow formulation of a more focused request. These strategies minimize impact on the data sources. Individual queries typically take milliseconds.

Having said this, for a data source that is already displaying performance issues in serving existing applications, adding TBI access could further impact performance. In general, this is similar to adding more users. On the
other hand, TBI can also help to relieve performance issues since ExploreSpaces provide data caching. With it, some users may no longer need to go to the underlying data source directly and can use pre-populated ExploreSpaces instead. Spaces can be refreshed on some regular basis, for example, daily or when new data appears.

Q5 How are you mapping the relational databases? Are you using W3C R2RML?

A5: Currently, we are using the D2R mapping language to expose the relational data to RDF world. In the future this may change, but such mappings are mostly “under the covers” and done automatically, so we don’t expect much impact when the change happens. If a data source already implements R2RML and exposes itself as RDF, then TopBraid Insight can work with it as if it were a RDF data source i.e., through SPARQL endpoint.

Q6: What are you using to map XML data sources to RDF?

A6: TopBraid uses Semantic XML which is a small vocabulary developed by TopQuadrant to expose XML hierarchies as RDF. As part of this process, when XML Schema is available, TopBraid Platform will utilize semantic descriptions from XML Schema.

Q7: Is an ExploreSpace the RDF merge of all the queries? Are you persisting it in a triple store?

A7: An ExploreSpace contains all information retrieved during querying and data exploration. It is persisted in a triple store. Users can keep the data in ExploreSpace as long as they want and/or they can clear spaces from the retrieved data. All the queries themselves are also stored and can be re-executed on demand to refresh the data.

Q8: Documents: indexing of free text to make searchable, annotatable?

A8: Yes, a ConnectSet can include document stores as a data source. For example, one of the customers uses 12 data sources: 10 relational and 2 document repositories with information such as patent filings. TopBraid Insight can use any available document metadata including search index. Many free-text documents are stored in document repositories and therefore have some amount of general meta-data associated with them. More detailed meta-data can be generated by processing the documents with text mining tools that extract meta-data from text. Documents processed in this fashion can therefore be better mapped to the unifying or a canonical model used by a ConnectSet. As a result, they will provide considerably more data-of-interest to users as they search and browse within an ExploreSpace.

Q9: Example of using OWL-based healthcare terminologies (ICD-11, SNOMED…) as a basis of data integration?

A9: Currently, we don’t have a demo showing the use of ICD-1 or SNOMED, but we are exploring these use cases with perspective customers. TBI could certainly be used for terminology mapping with the central model being the established terminology (e.g. ICD, SNOMED, Loinc, etc.) and the data sources using local models mapped to the central terminology. However, TBI’s capabilities extend far beyond those of terminology mapping, e.g. querying multiple data sources containing multiple patient records gathered from multiple disciplines (physicians, nurses, technicians, social workers, administrators, etc.) with each perspective using its own vocabulary…and more importantly, having its own view of all (or part of) a single central Model of Meaning. TBI would then enable highly informational queries to be made across the distributed data and thereby enable a more integrated and complete picture of a given patient to be developed.
Q10: Are ConnectSets SPINMaps?

A10: ConnectSets include SPINMaps by pointing to maps to be used. A ConnectSet pulls together all the data federation components. It identifies:

- What data sources participate in the federation
- What is the common or canonical model used by a ConnectSet
- How data sources are connected to the canonical model by identifying relevant SPINMaps
- What data entities in the canonical model are available for exploration in each ConnectSet and what properties to include in the full text search
- Optionally, how to identify the same entities across data sources using two alternative mechanisms - LinkSets and LinkMaps.

Q11: What are ConnectSets? Are they sharable? Crowdsourcable? [i.e. can data integration be distributed]?

A11: As per above, ConnectSets are defined using RDF. Each ConnectSet is stored as an individual graph.

The image below provides a pictorial overview of a ConnectSet. Canonical View Maps are implemented as SPINMaps. We have given them this descriptive name to explain that they connect data sources to the common model and to differentiate from “Link Maps”.

As with anything defined in RDF, ConnectSets are sharable and can be crowd sourced.

Q12: How are the schemas of these different databases aligned? SPIN? SPINmap?
**A12**: SPINMaps are used to connect data source schemas to the common or canonical model used for a specific ConnectSet.

**Q13**: Are predicates standardized? Or ad hoc?

**A13**: They can be either or both, this choice is up to the user.

**Q14**: B2B: integration of EHR (patient) data with bench data?

**A14**: We have not yet implemented TBI for this use case, but believe it is highly applicable to TBI capabilities. Any data source that has clearly-defined, non-ambiguous semantics (schemas) can be associated with a canonical model used by a ConnectSet.

**Q15**: Is the data space persistable?

**A15**: Yes, absolutely. It is persisted in a triple store and can be refreshed as needed.

**Q16**: How complex is your entire network data; looking for a quantitative measure of the network link data, so that you know how quick you will get your results?

**A16**: TopBraid Insight will use data source statistics. We also analyze queries against the available sources to plan the overall query strategy - when to send queries to sources in parallel or when to query one data source first and then query another one second after gaining information from the prior data source that will allow formulation of a more focused request.

**Q17**: Can the search box be used to construct complex Boolean queries over properties, such as ( (A = aaa) AND (T = ttt) ) OR ( (P in list {'P1', 'P2'})

**A17**: Not currently. This is possible as a future enhancement. Today, this capability is available in a different way – through the demonstrated parameterized form where users enter search criteria for different fields.

**Q18**: How does TQ/TBI address the “problem” of several equal search terms having different meanings? For example, the term ‘income’ has several different meanings in different domains and systems

**A18**: The answer to this question depends on the use case. For example, “income” field in one data source could contain before tax income. Another data source may also provide income data, but it is income after taxes. For a specific ConnectSet, users need to decide if they are interested in seeing data for both types of income and if it makes sense to bring it together. Some of the choices may include:

- Having two separate fields in the common model, “pre-tax income” and “after-tax income”, and map both fields accordingly.
- Use only one type of income data and ignore the other because it is not of interest.
- Use only one type of income, let’s say after-tax, in the common model. Use both data sources and figure out how to convert pre-tax income into after-tax income. For example, the data source may include tax percentage or some other information that could be used. Such normalizing calculations are supported by SPINMaps.

It is also possible that the meaning of income is the same in both sources, but the unit of measure is different i.e., different currencies. Again, SPINMaps will do on the fly conversion, as required.
Q19: How large can a connect set be and still provide efficient search times?
A19: If by large you mean a number of included data sources, we had a customer using 12 different data sources in a ConnectSet with good performance. As you can imagine, the answer depends on several factors. We’d like to hear more about your data sources and query requirements to provide you with a targeted answer. Also, keep in mind that TBI installation is not limited to a single ConnectSet. Multiple ConnectSets are supported to address interest of different user groups.

Q20: Not that we are volunteering, but when are you planning a real world reference site?
A20: Hopefully, soon. We are in discussions with a couple of customers regarding this and would welcome the opportunity to work with others.

Q21: How does it know how far down a “chain” of relationships to show in this ExploreSpace? How do you tune this?
A21: The depth of chaining is entirely user-specified with each search in the pane on the RH-side of the UI.

Q22: Are the data sources local or are you pointing to public endpoint?
A22: Both. Some of the data sources used in the webinar (CHEBI and CHEMBL) were local to our network hosted in relational databases, some (Drugbank and SIDER) were public endpoints.

Q23: Could you speculate about how easy it will be to implement a data mining tool on top of TBI? Say to find patterns spreading across different data sources?
A23: In general, TBI is a data exploration tool. We will need a more extended conversation to assess the use case.

Q24: Scalability: What are the constraints?
A24: Version one is constrained by a single active application server. There is no limit to the number of data sources.

Q24: How do you manage authorization between users and data?
A24: TopBraid integrates with LDAP and offers capabilities to manage access at a graph level. In the context of TBI it means that the access can be restricted to a specific ExploreSpace and/or to a specific ConnectSet. The latter means that a user will have access to all ExploreSpaces based on a given ConnectSet. If a user has access to an ExploreSpace, they will be able to see all data that can be brought into the space. This provides a lot of flexibility in deciding who should be able to see what data. While it is technically possible to also restrict access to individual data sources defined for a ConnectSet, we do not see it as practical since users may be receiving misleading search results.
Q24: Example of complex query, so show advantage of SPARQL over SQL?

A24: With RDF and SPARQL, relationships are explicit and directly accessible. As a result, SPARQL queries are considerably better aligned with users' conceptualization of a domain. For example, a query for medications typically prescribed for an ailment such as 'TB of vertebra' may look like:

```
SELECT ?medication
WHERE {
  ?diagnosis a example:Diagnosis .
  ?diagnosis example:name "TB of vertebra" .
  ?medication example:canTreat ?diagnosis .
}
```

In SQL this may look like

```
SELECT DRUG.medID
FROM DIAGNOSIS, DRUG, DRUG_DIAGNOSIS
WHERE DIAGNOSIS.diagnosisID=DRUG_DIAGNOSIS.diagnosisID
AND DRUG.medID=DRUG_DIAGNOSIS.medID
AND DIAGNOSIS.name="TB of vertebra"
```

In an SQL query relationships are unstated and derived from matching relevant primary and foreign keys of tables. It reflects the specific structure of the database and how the data is stored within it, not users understanding of their domain.

Queries that have to traverse a chain of connections are particularly complex in SQL and are very simple in SPARQL. For example, a user may want to retrieve all patients diagnosed with infectious or parasitic diseases. Diagnoses are organized in a hierarchy with many levels. For example, bacterial diseases, viral diseases, tuberculosis are all part of this category and are further divided into many nested hierarchical subgroups. The diagnosis for a particular patient can be described at different levels of granularity.

In SPARQL, the query may look as follows:

```
SELECT DISTINCT ?patient
WHERE {
  ?diagnosisGroup example:name "1. INFECTIOUS AND PARASITIC DISEASES (001-139)".
  ?patient example:hasDagnosis ?diagnosis .
}
```

Because SQL has no way to explicitly state a relationship and no equivalent of a "*" symbol which directs the query engine to traverse the link to the end, this would become a very complex and expensive query in SQL with many joins and unions to satisfy the possibility of a diagnosis for a particular patient to be described at a different level of specificity. Furthermore, the query would be possible only if a depth of the hierarchy was pre-defined, as relational databases can only store hierarchies of pre-set depth.

This brings us to a closely related topic – advantages of RDF data model over relational data model. When people ask what can be done with SPARQL that can't be done with SQL, often they are really asking about what can be done in RDF that can't be done with relational databases. This is an extensive topic in its own right and we encourage interested parties to take a look at TopQuadrant's training program.

In the context of TBI discussion, it is important to highlight one of the key advantages of RDF – the fact that it was designed specifically to make it easy to merge disparate sources of data. This is one of the reasons that the approach to federated query of integrated data within TBI is so flexible and powerful.
ON LINKSETS AND LINKMAPS

At the core of data federation is the need to match resources (instances) found in one data source to resources in other data sources. Since they will have a different identity in each data source, we need to tell TBI how to establish equivalency. With this, we will be able to run queries like "Find all movies portraying a person who wrote the book A Nation of Immigrants" when information about who wrote what book is in one dataset and information about who is portrayed in which movies is in another dataset. These mappings also enable the user to see merged information across all relevant sources when we, for example, click on the author of a book.

In some cases, it may be possible to specify the same URI creation policy across all data sources. For example, if each data source containing information about people has social security numbers and each data source about books contains ISBN numbers, then SPINMaps connecting the data sources to a canonical model can be defined so that these values are always used in the URI construction. As a result, data coming from different sources will merge automatically using the native capabilities of RDF. However, this is often not possible - some data sources may have social security numbers, but others may not. They may, instead, have customer IDs or just customer names and telephone numbers. In each case, a different logic or strategy may be needed to match data entities.

TBI accomplishes this matching using two complimentary approaches: LinkMaps and LinkSets. A ConnectSet may use one or both of these methods. LinkMaps identify which resources are the same by using functional schema-level mappings. LinkSets, on the other hand, contain explicit links between resources that are the same.