Originally appeared in:
October 2009, pgs 41-46.
Used with permission.
HYDROCARBON PROCESSING®

Service-oriented architecture simplifies data source integration

Here's how the approach helps refinery scheduling and also contributes to business-wide SOA adoption

K. SAMDANI, Infosys Consulting, Bangalore, India

The refinery scheduling system needs to interface with various heterogeneous data sources. Although the traditional point-to-point integration approach serves the needs, it presents several problems due to a variety of underlying platform technologies. Also, such a solution is not scalable. Applying the SOA approach helps automate the refinery scheduling process as well as contributes to building a business-wide services repository.

This article compares the traditional integration approach for the refinery scheduling process with the SOA-based approach and presents a conceptual architecture along with the benefits thereof. It also aims at bringing out how the SOA approach for individual projects such as refinery scheduling contributes to overall strategic SOA adoption by the refining business.

Introduction. The modern integration approach suggests SOA for the whole organization for strategic business transformation. The Forrester survey[1] indicates that broadly 70% of large businesses and nearly half of small to medium-sized businesses are into SOA and, more importantly, are by and large satisfied.

Adopting the SOA approach for refinery scheduling helps in two ways:

- The SOA approach develops a reusable scheduling services catalog that becomes part of an overall business-wide services repository.
- The SOA approach provides open standards-based integration of the scheduling tool with a variety of data sources.

The refinery scheduling process can be viewed as a set of reusable services. A service (for example, getting tank inventories) that is necessary for refinery scheduling can be used by other business processes such as yield accounting, plan vs actuals analysis, etc. Also, it can be used across other refineries. SOA also provides integration standards for interfacing with the various heterogeneous data sources such as historians, laboratory information management systems (LIMS), spreadsheets, etc. It eliminates platform dependence by using open standards-based service interface specifications. It enables organizing the scheduling process as various reusable services, thus contributing to developing a business-wide services repository.

Traditional approach to refinery scheduling. Refinery scheduling is largely driven by a scheduling tool. The tool generates schedules by processing data from a variety of sources. Unlike in the past, current refinery scheduling tools employ advanced mathematical engines to develop an end-to-end refinery schedule. These tools need data such as tank inventories, qualities, crude arrival and product dispatches, prices, etc. from heterogeneous data sources.

Fig. 1 depicts the traditional approach of point-to-point interfaces between the scheduling tool and data sources. Although it serves the data needs of the scheduling tool, several problems associated with the traditional approach are:

- Data source systems vary widely in underlying platform, integration capabilities, sophistication, etc. Some sources are spreadsheets/text files whereas others may expose Web services for retrieving data.
- In case the legacy systems (providing data to the scheduling tool) are replaced by modern systems, the interfaces with such systems are required to be replaced.
- Developing an interface for a new data source is almost a fresh effort. Reusability is minimal.
- These interfaces are tightly coupled with source systems. Hence, changes in underlying platform technologies of the source systems mean a lot of rework.
- In interconnected supply chains, data sources may be outside the organization. Supply chain partners have their own system

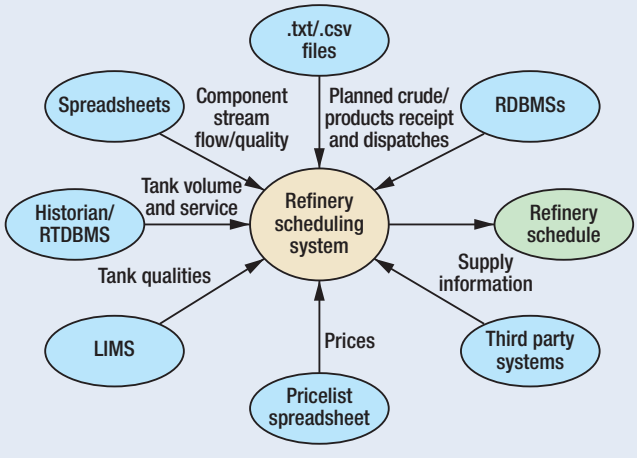

FIG. 1 Traditional integration approach.

upgrade/technology strategy roadmap. Any change to the underlying technology by them impacts the interface.

- For multirefinery organizations, the development and global deployment costs are high since there is little reusability.
- Maintenance/upgrade costs are high.
- Typically, these are not in line with the overall enterprise-wide IT strategy and hence, carry a lot of risks.

Point-to-point interfaces are typically implemented quickly. They serve tactical short-term objectives. Having selected a scheduling tool, the scheduler expects to start using it as soon as possible for obvious benefits of generating optimal and/or feasible production schedules without considering integration-related issues.

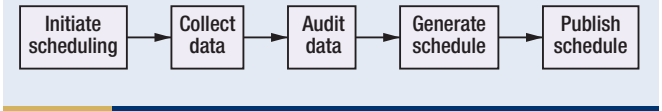

FIG. 2 High-level view of the scheduling business process.

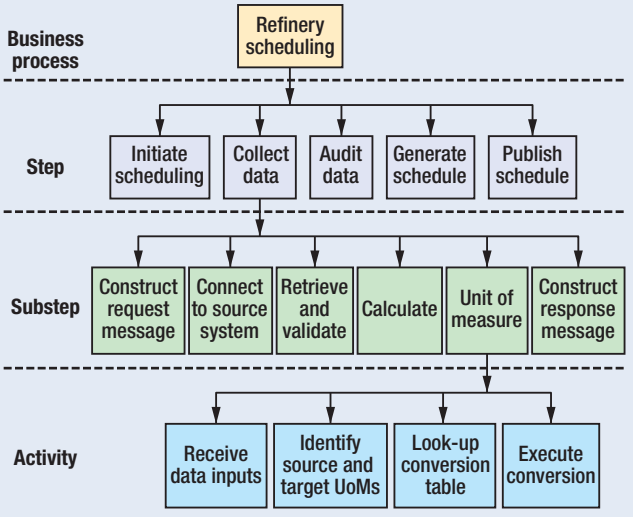

FIG. 3 Representative organization of refinery scheduling services.

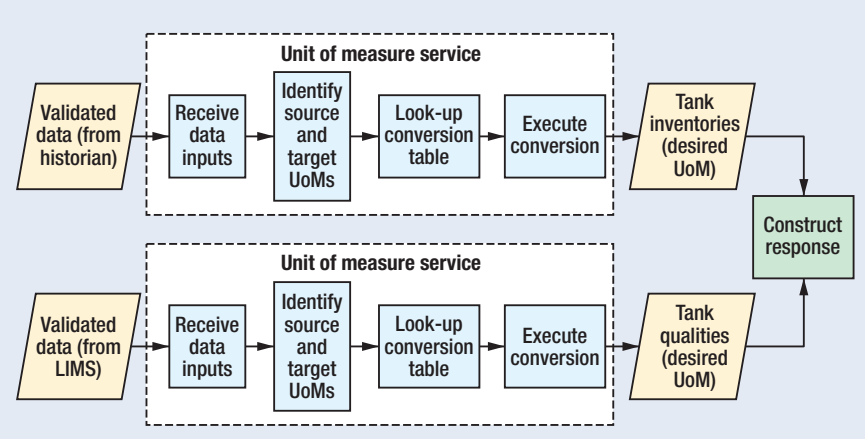

FIG. 4 Reusability of unit of measure service (substep level).

However, businesses are taking a holistic view of integration needs of multiple applications across the organization and developing integration standards. They expect a robust integration solution that considers reusability, is globally deployable across multiple refineries and reduces total cost-of-ownership.

SOA-based approach for scheduling. The principle objective of the SOA approach is to eliminate platform dependence and organize the business as a set of reusable services. In the refinery scheduling process, the primary service is to generate the refinery schedule. Generating the refinery schedule is achieved by performing several services. SOA decouples these services from source or target systems. Thus, it enables using a service as and when needed by any other service or application.

As shown in Fig. 2, the refinery scheduling business process can be viewed as made of multiple steps such as *initiate scheduling*, collect data, audit data, generate schedule and publish schedule.

Fig. 3 presents an example of how the refinery scheduling business process can be organized as various services. These services are classified into coarse-grained (step) and fine-grained services (substep and activity).

The refinery scheduling business process is divided into five key steps. The *initiate scheduling* step triggers the *collect data* step to get data from various sources such as historians, LIMS, spreadsheets, etc. The *audit data* step audits incoming data and shares it with *generate schedule*, that runs the scheduling engine and produces an optimal schedule. The *publish schedule* step publishes it.

Each step has been further divided into substeps. For example, *collect data* uses various substeps: *initiate*, *construct request message*, *connect to source system* through to *construct response message*.

Each substep can also be logically divided into activities. Fig. 3 provides an example of how the *unit of measure conversion* substep uses four activities.

The SOA approach expects such detailed organization of coarse-grained as well as fine-grained services. This enables developing a services repository. It is essential to consider how services can be independent of source and target applications while developing the services repository, thereby increasing reusability of services.

SOA enhances reusability of services. Systematic organization of services as depicted in Fig. 3 helps identify reusability and removes redundancies. Once organized, such a unique set of services representing the scheduling business process can be called by any application/business function within refinery operations as well as the broader enterprise. To maximize reusability, services need to be developed in a manner such that multiple applications/other services can use them. Creating application-independent services works very well for reusability.

Reusability of services can be demonstrated at multiple levels, i.e., at the substep or step levels or across the business processes.

Reusability at the substep level. *Unit of measure* is a substep within the *collect data step*. Fig. 4 demonstrates how the *unit of measure* service is used for converting input data units of measure from different data sources.

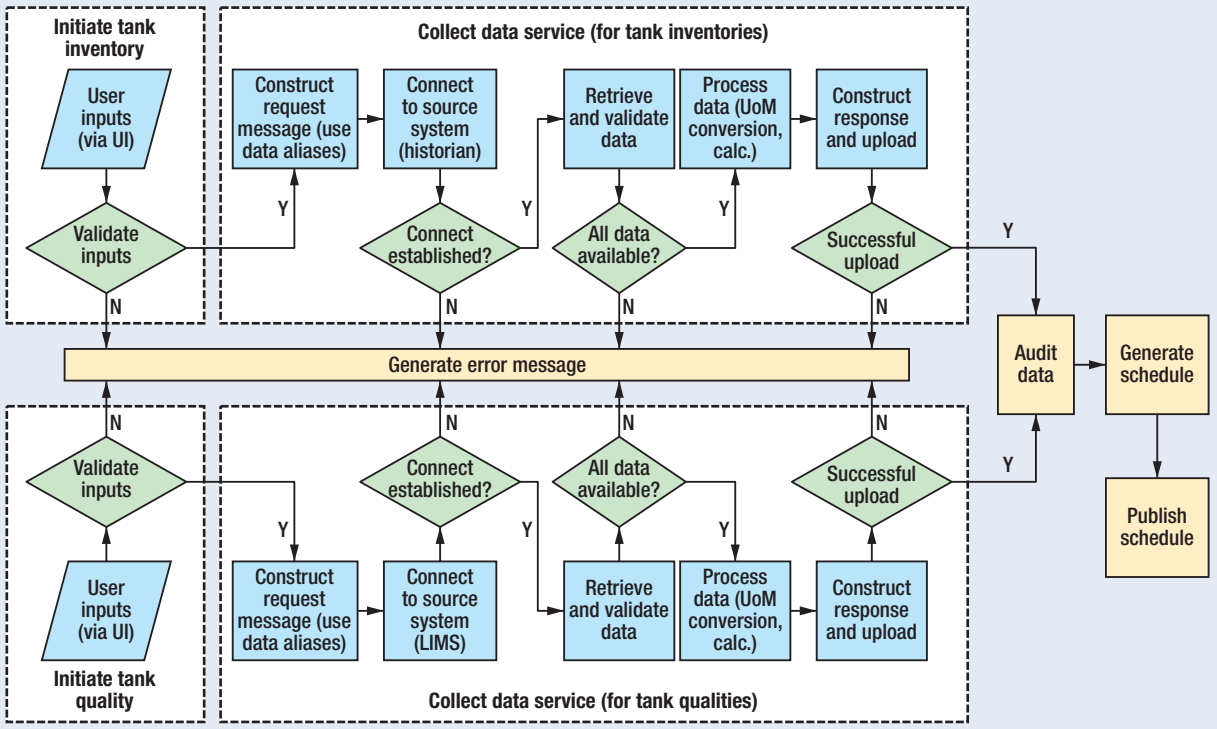

FIG. 5 Reusability of collect data service (step level).

The *unit of measure* service receives validated data from various sources (historian, LIMS, etc.). It is expected to convert the unit of measure as required by the scheduling system. It embeds four activities to convert units of source data to the desired units of measure. It does these activities in the same manner whether it receives tank inventories from the historian or tank qualities from the LIMS. Likewise, it can be used by any other service for converting units of measure of other data such as prices, crude arrival schedule, etc.

Reusability at the step level. *Collect data* is one of the five steps of the scheduling business process. It receives its trigger from the *initiate scheduling* step to get data from various sources and provide it to the subsequent step—*audit data*. Fig. 5 demonstrates how the collect data service is triggered by two *initiate service* triggers (*initiate tank inventory* and *initiate tank quality* triggers) to get data from the historian and LIMS. As shown in Fig. 5, the *collect data* service employs several substeps such as *construct request message*, *connect to source system* through to *construct response message*.

Irrespective of the trigger (whether *initiate tank inventory* or *initiate tank quality*), the *collect data* service employs the *construct request message* service to receive inputs and build the request message. In a *tank inventory* trigger, the input is typically a date. For the *tank quality* trigger, the input is a date range. The *collect data* service abstracts such differences and enables reusability for any such trigger.

Reusability across business processes. Fig. 6 demonstrates how the collect data service can be used for the refinery scheduling business process as well as the yield accounting business process.

Both of these business processes expect tank inventories. The *collect data* service employs substeps to get tank inventories. Therefore, it is available to be invoked independent of the business process (whether the scheduling process or yield accounting).

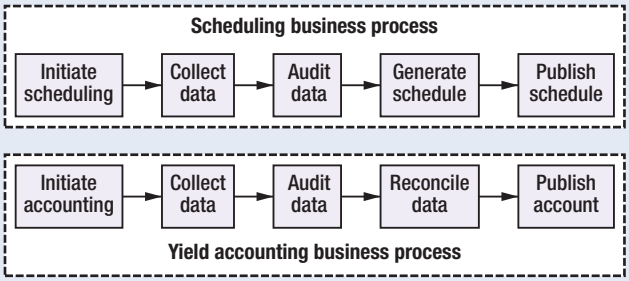

FIG. 6 Reusability of collect data service (business process level).

SOA-based scheduling tool integration. Refinery scheduling system integration with data sources is quite challenging since it presents a variety of platforms such as RDBMSs, historians, legacy systems, third-party systems (outside of the intranet), spreadsheets/files hosted on a local server, etc. SOA aims at eliminating platform dependence using open standards-based service interface specifications [such as Web services description language (WSDL[2])] and messaging protocols (such as SOAP, REST, etc.). Fig. 7 presents the traditional approach as well as the conceptual SOA-based approach for integration solution architecture for refinery scheduling.

As depicted in Fig. 7 and described in Fig. 1 earlier, the traditional approach provides point-to-point interfaces between data sources and the scheduling system. The conceptual SOA-based

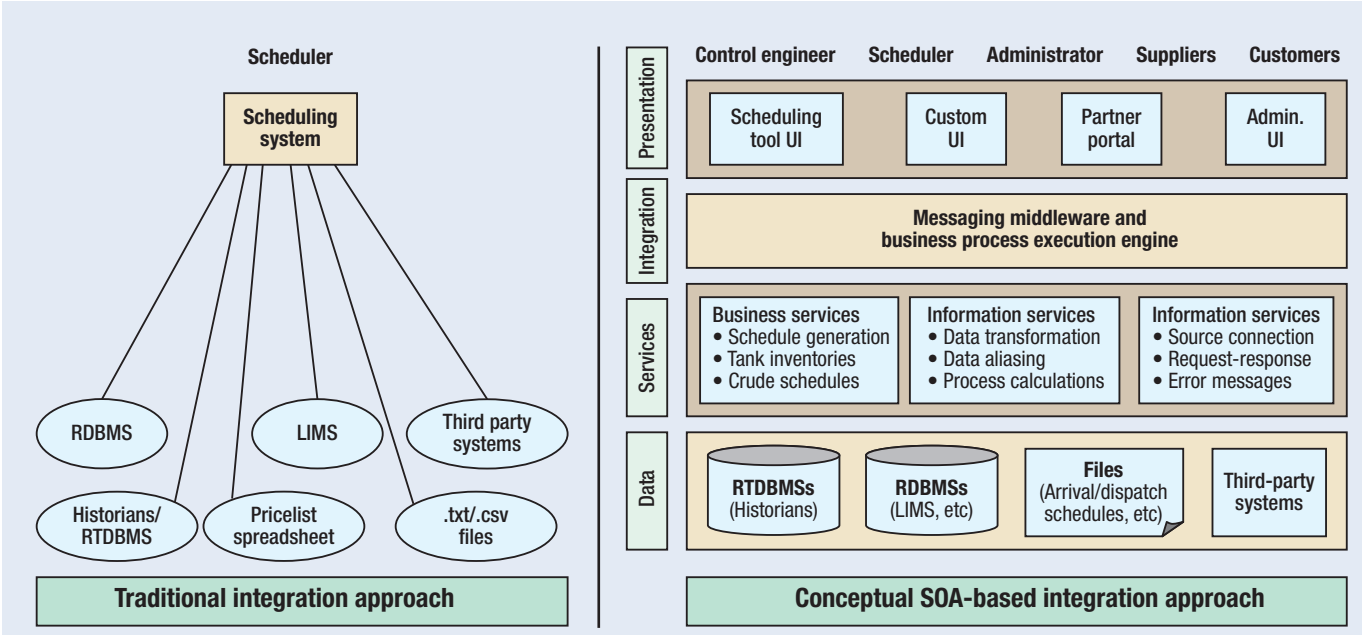

FIG. 7 Traditional and SOA-based integration approaches for refinery scheduling.

architecture presents four layers: presentation, integration, services and data. Key characteristics of each layer are:

Presentation layer: This layer provides data visualization capability for all concerned stakeholders. The presentation tier aims at providing a common user interface for all users. It delivers contextually relevant information to different users such as schedulers, control engineers, managers, partners, etc. It enables centralized authentication, authorization and context information sharing with integrated applications.

Integration layer: This layer manages scheduling process execution and integrates with various data sources. It provides features such as process management, services' orchestration, routing, security, logging and auditing. It can also help in enterprise-level integration needs. Several advanced technologies are available to enable the integration with a variety of data sources.

Services layer: The principal component of the SOA-enabled solution is the services layer. This layer decouples business logic from the presentation and data layers. It hosts services. These services are aimed at delivering business functionality, data management functionality and technical integration capability. SOA helps organize these services in a systematic manner (Table 1). The service categories are logically derived based on overall services' organization.

Data layer: This layer hosts data source systems. For example, RTDBMSs (historians) that provide tank inventories whereas typically the LIMS, which is an RDBMS-based system, provides quality data. Some of these systems provide direct database access while others expose Web services. Mapping a database to provide relationships among data from disparate sources is held within the data layer.

The SOA-enabled solution simplifies and automates executing the scheduling process and streamlines it in following steps:

- The scheduler triggers one or more services using the presentation layer.
- The integration layer responds by calling necessary services and orchestrates the process execution.
- Each service executes the desired functionality based on available inputs and provides outputs. Technology services enable connection services to data sources and retrieve data. Information services provide data management capabilities. Business services transform the data into desired output.
- The scheduling engine processes uploaded data and generates the refinery schedule.

To achieve the automation and simplified execution of the refinery scheduling process, the key enablers for SOA-based implementation are:

- Defining services
- Creating services repository
- Enabling all (data sources and destination) systems to participate in SOA approach.

These enablers lay the foundation for adopting SOA within the scheduling business process and overall refining business.

TABLE 1. High-level services categorization

Service category	Service description
Business services	These services help achieve the business objective of generating a schedule. An example business service is *collect data* which retrieves inventory data from the historian.
Information services	These services provide specific data management capabilities. For example, *unit of measure*.
Technical services	These services provide technical features and capabilities. For example: *generate error messages*, *connect to source system*, etc.

HYDROCARBON PROCESSING OCTOBER 2009

Benefits of SOA-based integration. Adopting SOA for the scheduling process is substantially beneficial over traditional approaches. SOA eliminates platform dependence by using open standards-based interface specifications. It enables connecting

SOA adoption helps not only end-to-end automation for the scheduling business process but also opens up opportunities for substantial benefits at the enterprise-level.

to any data source system. It enables identifying and organizing the scheduling process as a set of reusable services. Each service can be monitored easily. It enables better control over the entire scheduling process leading to continuous improvements. Some of the benefits are:

Greater reusability: The SOA approach develops a catalog of reusable services. For example: *Connect to source* system can be used by the *collect data* service to get tank inventories from a historian as well as to get tank qualities from a LIMS.

Enterprise-wide applicability: SOA decouples business logic from the presentation and data layers. A service can be used by other services within refinery operations. Such reusability can be at various levels such as substep or step, or even at the business process level. For example: The *collect data* service can be used by refinery scheduling as well as yield accounting business process to get tank inventories.

Reduced time-to-market: Reusability is just not within a refinery. It can be across refineries. For example, the *tank quality service* can be reused if all refineries have the same LIMS system. This favorably impacts the large-scale global roll-out programs.

Facilitates loose coupling: Due to loose coupling between applications and services, any changes in one application are isolated and do not impact other applications' functionalities. There are no point-to-point connections.

Reduces platform dependence: The SOA approach eliminates platform dependence. Virtually any data source application can be plugged into the architecture.

Enhanced collaboration: SOA enables tremendous collaboration not only within the refinery but also with supply chain partners such as traders, third-party crude storage organizations, customers, etc. They get a view of necessary information over the Web-based portal enabled by SOA.

Valero published a half-million-dollar savings in demurrage costs due to an enterprise-service-enabled application that improved visibility across business functions.[3]

Thus, SOA adoption helps not only end-to-end automation for the scheduling business process but also opens up opportunities for substantial benefits at the enterprise-level. **HP**

LITERATURE CITED

[1] Heffner, R., Vice President and Analyst for Forrester Research with ebizQ on "Current state of SOA adoption" in Oct. 2008.
[2] http://www.w3.org/TR/wsdl20/.
[3] Article in *CIOInsight*, July 2007.

Kailash Samdani works with Infosys Consulting. He has 16 years of experience in consulting and business development of IT-enabled advanced solutions to the hydrocarbon, chemicals and metal industries. He holds a B.E. (Hons) degree in chemical engineering from Birla Institute of Technology & Science, Pilani (India).