

New Developments in Information Technologies

By Weiss, L.W.¹

¹Sugars International LLC, 30 Glenmoor Dr., Englewood, Colorado 80113 USA
E-mail: WWeiss@SugarsOnline.com

Abstract

The development of Extensible Markup Language (XML) is leading to new inter-application communication that will provide more flexibility for software utilization in an enterprise. Initially, XML data is passed between programs in a manual asynchronous manner for programs that do not need continuous interaction; for example, importing factory data into the Sugars modeling program to provide on demand heat, material and color balances along with the net process revenues for the factory. In the future, Web services can be used to provide synchronous or asynchronous interactions using Web Services Description Language (WSDL) and SOAP for message passing in a Service-Oriented Architecture (SOA) to provide interactive communications and functionality between software applications used by a sugar company.

Introduction

The wide spread use of computers has proceeded at a dizzying pace in the 23+ years following the introduction of the IBM personal computer on August 12, 1981. The original operating system for all IBM compatible computers was DOS and most programs in the DOS environment could not talk to each other. People adopted a number of software applications that they used to cover different disciplines such as spreadsheets, engineering, project management, word processing, database applications, accounting, graphics design, etc. In general, these programs were dedicated to a specific task that once completed the program was closed and another program would be opened to accomplish another task. Originally, information from one program could only be transferred to another program by manual input, or some sort of data export and import using a format that was mutually agreeable to both programs. Other programs might have different formats and unless the programs were designed for this different format, they were not capable of communicating. Some attempts were made to introduce software that integrated several programs so that information could be shared, but most of these programs lacked sufficient capability to find widespread use. With the introduction of Microsoft Windows 3.0 in May of 1990, some sharing of information could be done, though it was limited. The release of Windows 95 in August of 1995 introduced new techniques for program communication through the Component Object Model (COM) and variations of COM that allowed software interaction so that data could be passed between programs and procedures in one program could be used by another program.

The ultimate goal is to have software programs that can exchange information in much the same manner as humans do with normal discourse. The problems with software data exchange are similar to human interactions. Humans have to work with different languages, dialects, knowledge level, application, and lines of communication. The same sorts of problems exist for

software. However, new developments in the exchange of data are occurring that may finally resolve some of these difficulties. The basis of these new developments is a method for sharing information between programs that is called Extensible Markup Language (XML). It is similar to HyperText Markup Language (HTML) for designing Web pages, but it is more versatile. Programs can now be designed with an interface called a Web Service that uses XML to communicate with each other. The communication uses another XML technology called SOAP to transfer XML information between programs. Finally, the combination of these technologies into a system of different programs that exchange information and procedures is done with a Service-Oriented Architecture (SOA) that enables distributed computing. This paper gives a brief discussion of these technologies.

Extensible Markup Language (XML)

XML is a language that defines knowledge, data and metadata about applications. The structure of an XML file is composed of *elements*, *attributes* and *values* such that data can be handled in different ways by different programs. XML is more flexible than HTML because the elements, attributes and values are not predefined; instead, they can be defined to be anything. HTML is used mostly for presentation, such as with Web pages; whereas, XML separates the content from the presentation, and it can be used for data exchange. Further information about XML can be obtained from the web sites: <http://www.w3.org/xml> and <http://www.w3schools.com/xml>.

An XML file is not difficult to read and its basic concept is not complicated. For example, the portion of an XML file defining a temperature of a flow stream could be written as follows.

```
<Property Name="Temperature">
  <Value>93.4</Value>
</Property>
```

In this small example, the element is "Property", the attribute is "Name" with a value of "Temperature" and the "Value" is, of course, 93.4.

Elements can be nested such that the property in the above example could be tied to a flow stream or equipment (e.g., vacuum pan); hence, the XML file to define temperature and dry substance values for the massecuite leaving a pan might look as follows.

```
<Station>
  <Equipment>Pan2060</Equipment>
  <Property Name="Temperature">
    <Value>78.0</Value>
  </Property>
  <Property Name="DrySubstance">
    <Value>93.0</Value>
  </Property>
</Station>
```

A complete XML file for passing data between programs contains other information about the file and a reference to a *schema* that is used to validate the file. The schema is an XML file that uses the Extensible Schema Definition (XSD) syntax (see <http://www.w3.org/XML/Schema>). It defines the namespace for the names of elements and attributes allowed in the XML file. For example, the namespace for the above XML example would include the elements named Station, Equipment, Property and Value and the Name attribute. But, if the XML file contained “Field” as an element, it would not be allowed unless it was added to the namespace.

Schemas define the structure of the XML data and verify the correctness of the names used for elements and attributes and the data types used for the values. The data type for the Value element for temperature and dry substance in the above example would be limited in the schema to be a real number or “float” type. As shown in the example, the XML data file for the temperature and dry substance values only contain content; that is, it does not contain the method for presenting the content (for example, the font used on a Web page, or within a document, or where it should be placed). Instead, an Extensible Stylesheet Language (XSL) or Extensible Stylesheet Language Transformation (XSLT) is used for doing the presentation. XSLT can transform XML into HTML for display in a browser; format it using the data value (for example, bold text and/or color if the value has a special meaning); sort it; prepare it for display on a monitor; transform it to audio; or, transform it for other methods of presentation.

Microsoft Office 2003 is an application that has many new features for transferring XML information between different Office applications and for editing XML files. Word 2003, for example, can display and edit XML documents while checking the document against the schema. In fact, Office 2003 was one of the first major applications to make full use of XML. Documents can be created as XML documents on non-Windows platforms and then loaded directly into Office 2003 applications. And, conversely, Word 2003 and Excel 2003 can save documents in XML for loading into other programs.

Importing data to and exporting data from a program can be done manually in an event driven or on demand asynchronous mode. That is, data in XML format can be exported from one program to a file that is then read by another program and imported into that program for processing. The program importing the XML data will process the imported file, parse out the data from the file and then employ the imported data by using program code designed specifically for the application. This is the method used by the new Advanced Monitoring System (AMS) add-on feature of Sugars™ as shown in Figure 1 below¹.

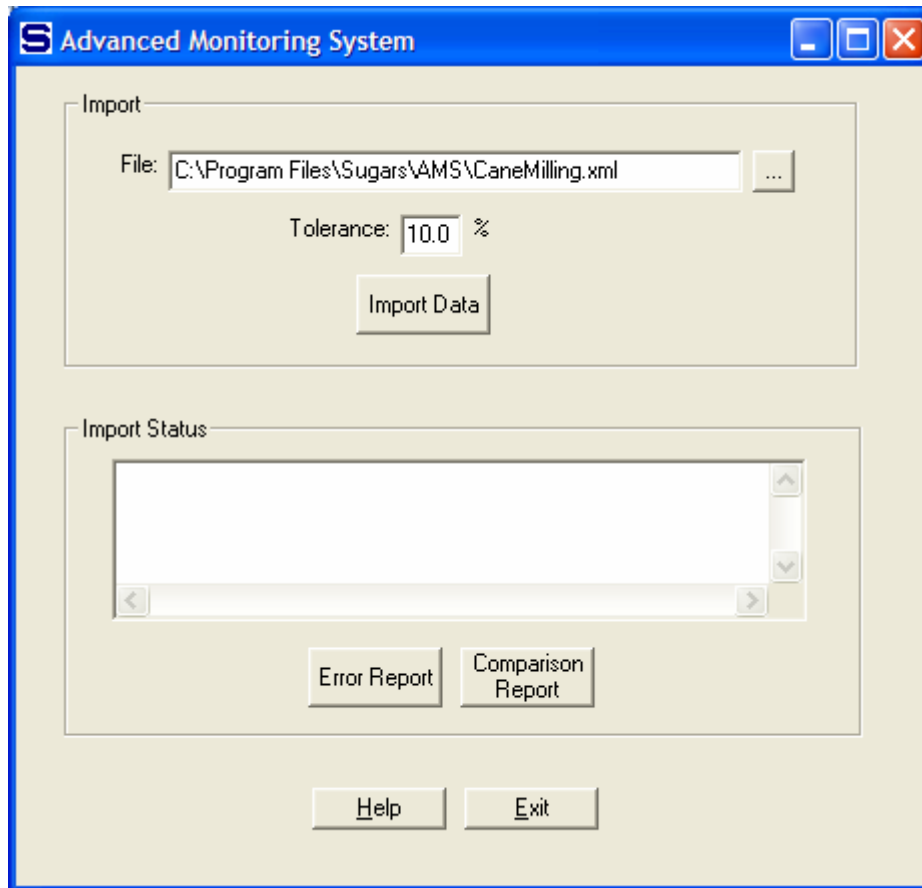


Figure 1. XML data file import screen.

Data from the data acquisition system is saved in an XML file (e.g., “CaneMilling.xml”) and these data are imported into a Sugars model of the factory using the window shown in Figure 1. The data are filtered for a tolerance allowance (10.0% as shown) and any errors that occur during the import are given on an Error Report by clicking on the Error Report button. The Comparison Report shows the old values with the new imported values and the change in the value as a percentage. Also, data from the data historian can be sequenced such that it coincides with the movement of material through the factory. Because the import is XML data, any data acquisition or management information system that can export data into a properly structured XML file can be used to update the data in a Sugars model. That is, the data import feature for the Advanced Monitoring System is not tied to any particular data gathering system.

Once the data is imported, it can be displayed on the flow diagram of the model as shown in figure 2 below.

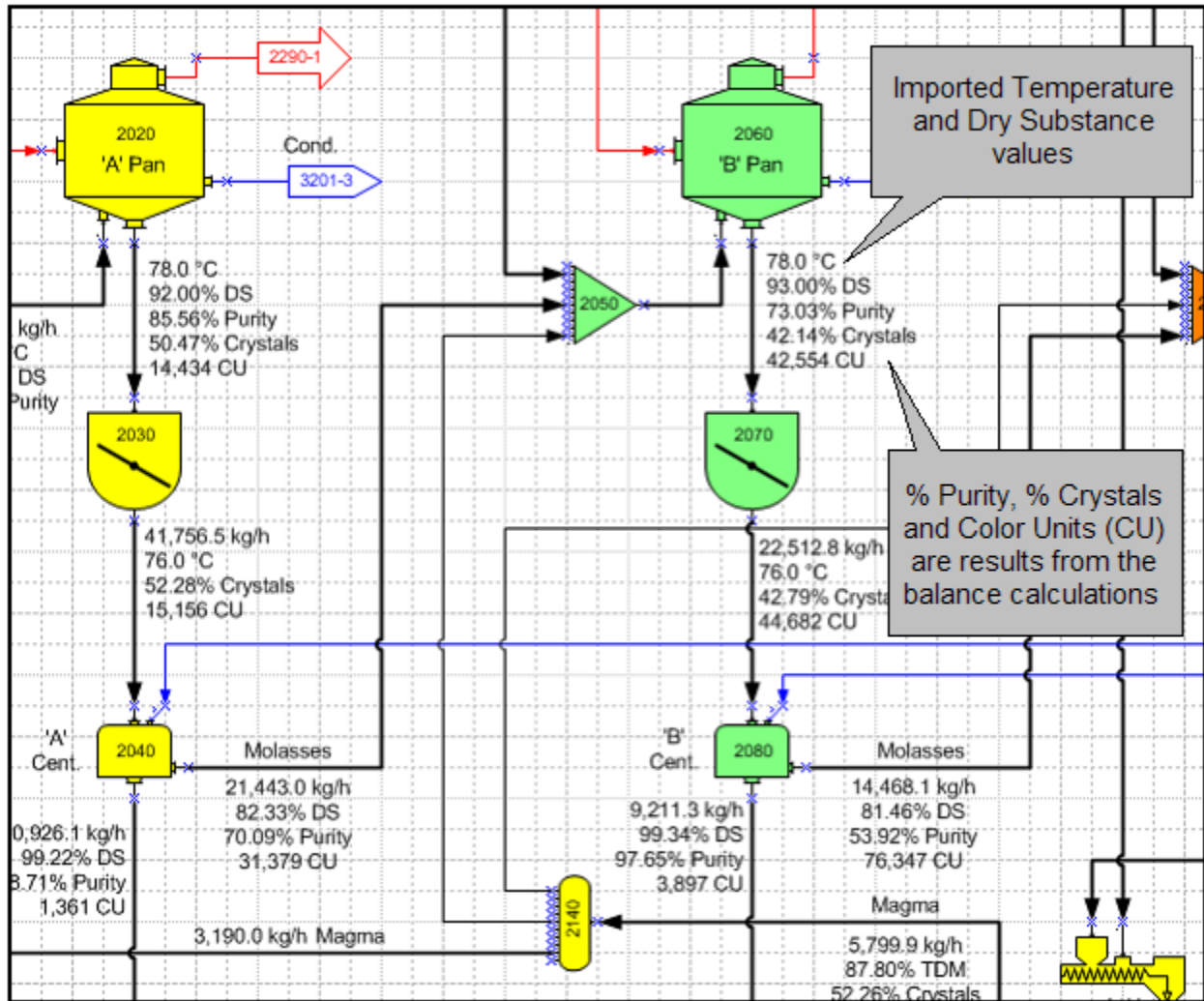


Figure 2. Flow diagram showing imported and calculated data.

The Sugars program then uses the newly imported values to rebalance the model and give the heat, material and color balances, along with the net process revenues. Details about all of the flow streams and the overall performance are also provided by the balance. This provides information by inference about flow streams that are not measured. In the current implementation of the AMS, a reply is not given from the modeling program back to the data acquisition or management information system; however, future development can add this feature such that the management information system could send data to the modeling program and receive replies. Replies from the modeling software could contain information for optimization of the process; or, equipment that needs adjustment, repair (e.g., centrifugal screen replacement), or cleaning (e.g., scale removal). Development of this feature will use a Web service interface with messages between the programs using SOAP.

Web Services and SOAP

Web services provide interfaces to programs to enable them to accept data and return data to sending programs if a response is requested. They can connect programs running on any platform by using industry standard communication protocols. The data is usually in XML format; however, Web services can also use data in other formats. Normally, one application converts data into XML in the form of a message and sends a request to another Web service on the network (local or internet) or another application on the same machine. The application will receive a reply as XML data if it requests a reply. Web services define the data, how to process the data and how the data is moved into and out of the software application that it serves. The software application can be written in any language; hence, Web services are not limited to certain types of applications. This results in “seamless computing” to create a fluid flow of information between applications, devices and businesses.²

The complete deployment of Web services requires the use of SOAP for communications and Universal Description, Discovery, and Integration (UDDI) for locating services on the internet if services are to be used across the internet and they are needed from other external applications (see the web site www.uddi.org). The characteristics of a Web service for an application are defined in a Web Services Description Language (WSDL) schema. WSDL is an XML document that provides a framework for describing the data that is passed in messages and how it is encoded and decoded to and from virtually any application. It allows programs from different platforms to understand their protocol requirements with very little or no user involvement.

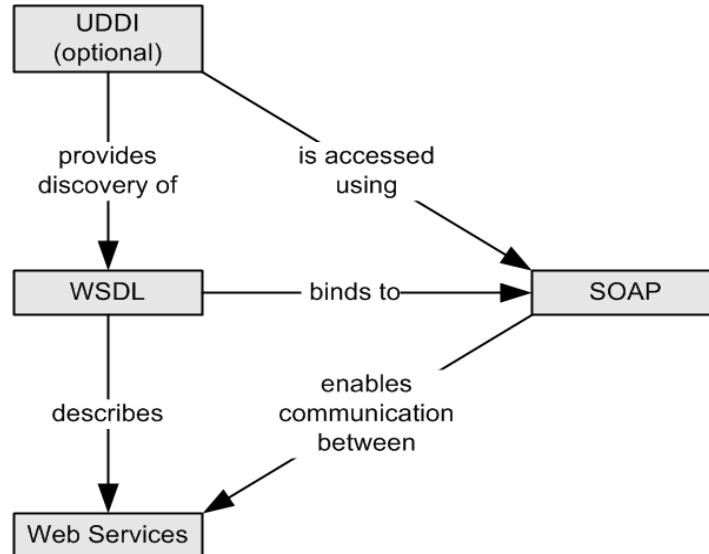


Figure 3. Web services specifications relationship.

Figure 3 shows the relationship between the various specifications that are used for Web services³. XML is the foundation for each of the specifications, and SOAP is the glue that ties them together. SOAP was originally an acronym for Simple Object Access Protocol, but as of the SOAP specification 1.2, it no longer stands for anything other than a generalized XML

messaging framework over various protocols for use between applications. A SOAP message has three major parts: envelope, header (optional) and body. The envelope marks the start and end of the message. The header can contain one or more blocks with context information about the message or the services for the message. The body contains the message itself⁴. Further information about SOAP is available on the web site: <http://www.w3.org>.

Interoperability between applications using XML, Web services (WSDL) and SOAP is still evolving, but the Web Services Interoperability Organization (<http://www.ws-i.org>) has recently released the WS-I Basic Profile 1.1 that delivers a set of rules that can be followed by vendors to ensure interoperability is built into their software⁵.

Service-Oriented Architecture (SOA)

A Service-Oriented Architecture is a method of architecting applications such that they interact with each other using services. Services are interactions which have the following characteristics: (1) they allow the applications to be autonomous from each other, (2) their boundaries are explicitly defined by a schema, and (3) their use is defined by a policy. Because they are autonomous, the individual applications can change without requiring another application to change. The only description needed to implement the service in different applications is given in the WSDL schema. All information about the availability, security, interaction, and communication are defined through an XML policy; and hence, the services can be negotiated automatically by the applications involved in the transfer of data and functionality.

The architecture involves designing which services are provided by which applications and when and how they are available. Any application can expose a service and they can reside anywhere on the network. Each service is listed in either a public or private registry so that it can be located by the other services. Some services would be accessed over the internet; for example, commodity pricing. Others would only be used internally to provide information specific to the needs of the enterprise. The SOA would be designed to access the necessary services from the appropriate applications to provide the operation with the required functionality.

As an example, consider a sugar factory process data acquisition system supplying a modeling program with current data and the modeling program providing the management information system with complete information regarding energy usage (heat balance), overall factory efficiency, sucrose losses, net process revenues from updated products pricing using a Web service in the sales department software, and information transferred back and forward between the factory maintenance program and the modeling program. Knowledge from each of these programs could be shared with other programs to provide information for optimizing the process (energy consumption, other consumables and product yield), control the maintenance of equipment and instrumentation, while maximizing revenues. All of this would be done on an open system that is interconnected using XML, Web services and SOAP.

References

¹Sugars™ is a process modeling and simulation program for the sugar industry provided by Sugars International LLC (web site: <http://www.SugarsOnline.com>). The Advanced Monitoring System (AMS) is a joint development of Sugars International and IPRO Industrieprojekt GmbH (web site: www.ipro-bs.de).

²Information Week, December 1, 2003, page 21.

³*Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services*, Erl, T., Prentice Hall, Copyright 2004.

⁴*Understanding Web Services*, Newcomer, E., Addison-Wesley, copyright 2002.

⁵PC Magazine, May 18, 2004, Lizschutz, R.P., page 76