


SiLA: Basic Standards for Rapid Integration in Laboratory Automation

Journal of Laboratory Automation
17(2) 86–95
© 2012 Society for Laboratory
Automation and Screening
DOI: 10.1177/2211068211424550
<http://jala.sagepub.com>
 SAGE

Henning Bär¹, Remo Hochstrasser², and Bernd Papenfuß³

Abstract

A standard (SiLA, Standardization in Lab Automation) that focuses on the connection between sample processing devices and a software system for automation gains acceptance. This article reports about the creation and contents of the first set of standard specifications developed during the first 2 years. These specifications—named Device Control and Data Interface Specification, Common Command Dictionary, and Data Capture Specification—describe how devices are connected to a software controlling the interplay of the devices, the command sets for various device classes, and the structure of result data such as data generated by microtiter plate readers. A section about SiLA-compliant products and pilot projects using SiLA-compatible devices for system integration gives an idea about the acceptance of the standard in the marketplace.

Keywords

standardization, laboratory automation, high-throughput screening, drug discovery

Introduction

Automation in general is a cornerstone of many of today's technological achievements. Various industries rely on automation to meet the demand of goods at consumer-friendly price levels, such as the car, packaging, and electronics industries. Other industries, such as solar module fabrication, have lagged these developments for a long time but recently have caught up.

Also in the pharmaceutical industry, especially in the drug discovery environment, the past 10 to 15 years have seen a tremendous development of a growing number of sophisticated systems for automated sample storage, sample replication, sample delivery, and ultra-high-throughput screening. In recent years, automation has reached new areas such as secondary screening, metabolism screening, toxicology, perfusion systems, polymerase chain reaction, and genome and proteome analysis. In most cases, these integrated laboratory automation systems are located in centralized service facilities.

In the traditional drug discovery laboratory environment, the extent of integrated automation is still limited. Liquid-handling robots and analytical instruments are widespread. On the other hand, such devices typically operate as standalone units. Sometimes a simple addition of one or two devices would enable more continuous and efficient operation. Different factors hinder a cost-efficient and rapid integration of a few devices to fully automated systems.

The high cost, the long implementation times, and the inflexibility of such systems restrict the penetration into smaller laboratories. In addition, the collected data are typically stored in proprietary formats such that visualizing and analyzing data from various sources can be complex and cumbersome.

In the field of drug discovery, an automated lab is set up by devices for sample preparation and analysis, such as incubators, pipettors, dispensers, or plate readers. These devices are connected to a software system that automates workflows through the setup. Such a setup without the use of a standard has many disadvantages. The different types of interfaces, such as USB or RS232, and their different physical properties make it difficult to connect to the controlling PC and in most cases require additional hardware. Serial connections are exclusive point-to-point connections and can only be used by one pair of partners. They have to be disconnected if a third party wants to access the device.

¹Infoteam Software AG, Stäfa, Switzerland

²F. Hoffmann–La Roche, Basel, Switzerland

³EQUIcon Software GmbH Jena, Germany

Received May 1, 2011.

Corresponding Author:

Henning Bär, Infoteam Software AG, Laubisrütistrasse 44, Stäfa, 8712, Switzerland

Email: baer@infoteam-software.ch

Modern networking technologies enable nonexclusive access to devices. The variations on the software side are even more detrimental. Devices with the same or similar functionality offer totally different command sets, and their temporal and structural behavior is vastly different. In addition, states of the devices are differently accessed and described for each device. Furthermore, the error messaging and the error-handling capabilities of the devices vary enormously.

To address these issues, in 2008, the SiLA Consortium (SiLA, Standardization in Lab Automation), consisting of device suppliers, integrators, and pharma companies, was founded with the goal to standardize network protocols in lab automation. This simplifies and speeds up the integration of laboratory devices into automation systems for the drug discovery industry, enables small integrations in a wider range of laboratories, enables fast replacement of devices for enhanced performance or new technologies, and reduces integration, support, and maintenance cost.

The SiLA standards are based on state-of-the-art interface technology enabling multiuser access to devices on simplified, easy to understand commands and on unified data formats. They are, however, not completely rigid and restrictive so that special functions and requirements can easily be accommodated within the framework of the standards.

Related Work

Standardization is a common approach to alleviate such problems. Different standards for laboratory automation have been developed over the past decade.¹

The American Society for Testing and Material (ASTM) standardized the communication of process instructions under the title ASTM 1394.² However, this standard focuses on the communication of a lab information system that manages information and data received from the laboratory. In addition, the protocol allows instructing complex tests, so it can be seen as a higher level of abstraction. The individual steps of devices, such as initialization or their error handling, are not targeted by ASTM 1394.

Health Level Seven (HL7)³ is a comprehensive set of standards of the clinical diagnostics. Besides other things, its chapter 13 of version 2 addresses the control of medical devices. However, there seems to be no straightforward way to describe the behavior of devices, such as a state machine. Version 2 of HL7 uses an outdated way to split different data values by vertical bars. Alternatively, an Extensible Markup Language (XML)⁴ can be used, but the structure is very cumbersome because XML hierarchy is avoided to allow conversion to the bar notation. Version 3 of HL7 is extremely complex, which results in no relevant usage in the industry since its publication in 2005.

Laboratory Equipment Control Interface Specification (LECIS)⁵ is a standard that focuses on device control. It is

platform independent, describes the behavior of devices by means of a state machine, and has only minor technical drawbacks. However, it has not been used since its publication in 2002. We assume that the reason is the lack of commitment of pharma companies to the standard and maybe also the somewhat outdated technical level of the LECIS standard.

For the representation of data, the Analytical Information Markup Language (AnIML)⁶ is an XML dialect that allows expressing analytical data according to a very generic XML schema. AnIML is implemented by a handful of companies providing analytical instruments and applied by a few pharma companies and academic research institutions, but it is not yet widely implemented.

In addition, many manufacturers of instruments have developed their internal standards, but none has yet been made publicly available or has been accepted by the community. Also, many integrators have developed their own internal driver libraries for a wide range of devices. This repetitive work could be eliminated with a standardized interface.

SiLA: Standardization in Lab Automation

SiLA was founded to address the needs for the interfacing and integration of laboratory devices used in the drug discovery industry. Therefore, the SiLA Consortium is based on important companies in the fields of drug discovery and lab automation. Strategic goals of the SiLA organization for standards development cover a wide range of topics that also include interfaces to laboratory information systems, remote access to automated systems, and data formats for high-content data. A natural starting point, on the other hand, was the standardization of the interface for devices.

With a vendor-independent interface definition that applies to all devices, it should be much simpler to set up flexible and modular integrated laboratory automation systems performing various tasks for the drug discovery researcher. In addition, such integrations will be much more time and cost efficient and easier to understand for the researcher and the technician. The usage of the identical interface for all devices will also increase the flexibility and the modularity of the integrated laboratory automation system, enabling fast replacement of devices so that new developments and technologies can easily be integrated in existing systems. Although similar approaches are generally referred to as “plug and play,” SiLA has not that ambition but certainly aims in the direction.

Figure 1 shows a typical setup of such an integrated laboratory automation system. The Process Management System (PMS) controls the devices integrated into the system. It resides in a computer that is connected to the devices by a hardware interface. Most commonly, this interface is a serial port connection. In more modern devices also, other interfacing technologies such as USB, CAN-bus, or

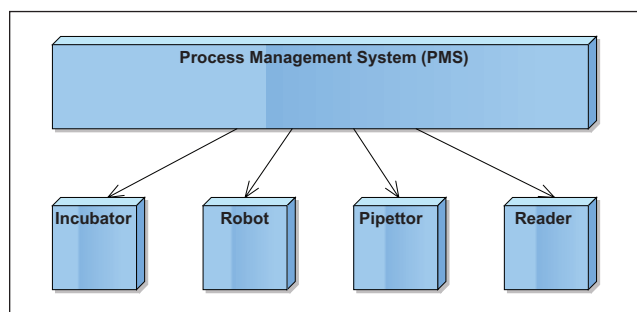


Figure 1. Integrated laboratory automation system with controlling software interfaced to devices.

Ethernet are used. These hardware interfaces enable the PMS to communicate with the devices through commands, and the devices reply by messages sent back to the controlling software.

The SiLA Standards

In this section, we describe the Device Control & Data Interface Specification that states how the communication between PMS and devices has to take place in general. Afterwards, we focus on the Common Command Dictionary, which defines the command sets for different device classes. The section finalizes with a description of the Data Capture format that can be used to describe result data of the devices.

Device Control & Data Interface Specification

The basic standard developed by the SiLA organization is the Device Control & Data Interface Specification. It describes how devices shall be connected to a PMS both from the hardware and from the communication protocol side. The detailed description below first shows how SiLA suggests adapting standard Web services to the needs of lab automation environments. Afterwards, the asynchronous communication with the different SiLA event types is described. Then the behavior of the devices, which is specified with state machines, is explained and, finally, how it handles the basic networking and wiring.

Web Services. The intent of the communication is to allow the PMS to start processes on the devices and to question the device about its current state. This shall be achieved by remote procedure calls to the device and by analyzing the corresponding responses. SiLA specifies the use of the Simple Object Access Protocol (SOAP)⁷ and the Web Service Description Language (WSDL)⁸ for such communications. These technologies are both based on XML.⁴ Each device provides a complete description of its command set using WSDL. Typically, the command set does not change once the

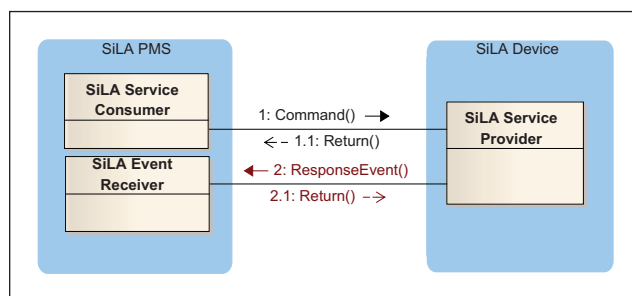


Figure 2. Asynchronous command invocation. PMS, Process Management System; SiLA, Standardization in Lab Automation.

device is installed. Therefore, the PMS needs to download the command set from each device only once. Then it can express a command, including its parameters, in one SOAP message, which it sends to a device. The device will interpret the SOAP message and start working on the command. SOAP and WSDL offer advantages in comparison to other technologies. They are independent of any platform or programming language, they offer type safety, they can easily be used through firewalls, and many tools available support their use. Long transmission times for huge amounts of data can be avoided by the use of HTTP compression. The process of compression can be done in a streaming manner, which means that a device sends a data block the same time as it already compresses the following data block. This approach avoids additional time for the process of compression.

Asynchronous Communication with Events. The processing times of the different commands a device offers may vary considerably. They might finish within less than a second or might take processing times of 20 min or more. Apart from two exceptions, SiLA specifies asynchronous communication for all commands to address these varying command execution times. **Figure 2** shows an according communication diagram between a PMS on the left and an arbitrary device on the right.

The PMS consists of a Web service client and an event receiver. The communication between these modules is not shown in the figure. For investigating the communication of a device, only one module is of interest for now. The communication starts with the client module of the PMS sending a command (1) to the service provider. The service provider acknowledges (1.1) that it received the command and starts processing. After finishing the process, the device sends a response event (2) back to the event receiver component of the PMS, which in turn acknowledges (2.1) the reception of the telegram to the device. In step 1.1 and in step 2, the device provides a special type that includes a return code. By this return code, the device can indicate success or failure.

Besides the response event, SiLA specifies further events to address other needs of typical lab automation setups.

Some devices produce measurement results. They can be sent en bloc via the response event at the end of the measurement. Alternatively, the data might be split into small units such as a measurement for each well. Then each result can be sent with a single data event. So for a 96-well plate, the measurement results can be split into 95 data events and one response event. To enable such a separation, the contents of data events and response events are structured alike.

The status event allows informing a PMS about changes in the state of a device that occur independently of a command sent to the device. Status events can be used as source of information about abnormal situations, such as an over-temperature or low liquid level warning.

The last defined event is the error event, which is evoked if a recoverable error occurs, such as missing liquid for a dispense procedure. The device can request a decision from the PMS on the way to handle that error. The user may decide between options such as dispensing less volume, refilling the consumable bin first, or choosing another liquid. All these options are sent to the PMS in the message body of the error event.

The WSDL of the device automatically provides information about available methods or commands and their parameters. Additional information relevant for the execution of the commands, such as ranges or units, should also be included. In addition, default values can be incorporated.

SiLA enforces an XML annotation of the WSDL documentation tag and provides the corresponding XML schema. Because the asynchronous communication behavior hides the definition of the result data totally from the Web service definition, the format and type of result data are defined also in the WSDL documentation tag. There is the possibility of having empty responses, sets of parameter/value pairs, and complex data types represented in an XML structure.

State Machine. SiLA describes the behavior of a device by means of a state machine. To cover the behavioral complexity of all possible devices, a simple diagram, as shown in **Figure 3**, is not sufficient. Multifunctional devices, such as pipetting robots that enable parallel processing and command queuing, require a more complex description. Therefore, subdiagrams are defined that also allow realizing these complex behaviors and also show how to implement smart, user-assisted error handling for recoverable errors. The state transitions *Pause* and *DoContinue* support the error handling and increase operator safety.

Figure 3 shows the main state machine. The state *AnyState* is a placeholder for any other state. All state transitions are equal for all devices and therefore must be realized by every device. The commands that enable these state transitions are thus required for every device. This is why they are called *mandatory commands*.

When a device is switched on, it resides in the state *Startup*. The first command that needs to be issued is the *Reset* command. With this command, the PMS can transfer

important data such as the address of the event receiver to the device, which is necessary for the correct operation of the integrated system. After the *Reset* command is successfully executed, the device is in the state *Standby*. Next the command *Initialize* has to be performed. That command leads to the state *Initializing*, which is left automatically toward *Idle* when the device has finished its initialization procedure. Now the device can accept commands that trigger processing on the device, such as *Read* for a microtiter plate reader. Such a command will result in a switch to the state *Busy*, and when the reading process is finished, the state machine will switch back to *Idle*.

The state *Busy* actually is an entire state machine itself, the busy state machine. The busy state machine again makes use of an internal state machine, which is called an asynchronous processing state machine. The tasks of these state machines are to manage parallel executions by means of multiple instances of the asynchronous processing state machine, to handle queuing of commands, to organize syntax and range checks of the commands, and to handle pausing of commands and event sending. The latter, more complex state machines must be implemented only when the device is capable of performing such tasks. For many simple devices, the uncomplicated state machine of **Figure 3** will be sufficient.

Network Specifications. A standard network connection using Ethernet is defined in the specification to allow connecting the devices to a PMS installed on a standard computer. The use of Ethernet connections gets rid of former restrictions commonly encountered with the standard interfacing methods, such as limited cable length, limited number of ports, or the risk of an RS232 interrupt blocking an entire automated workflow.

To support the integration of legacy devices that do not comprise a native SiLA interface, SiLA allows the use of SiLA-compatible software drivers that can be addressed by the PMS using the localhost [rfc2606] interface. In such cases, the driver is installed on the same computer as the PMS software, and the communication to the device can be established by any hardware interface.

Common Command Dictionary

A device can execute different tasks that are requested, scheduled, and controlled by the PMS. To start such tasks, the PMS sends commands to the device. For many currently available devices, the implemented commands can be cryptic with no obvious meaning to the programmer or integrator. Often the available commands are on a quite low level and therefore do not allow efficient and fast programming. One goal of SiLA is to replace the command names with meaningful and descriptive names. SiLA is also aiming to only define commands at a convenient level of complexity and usability.

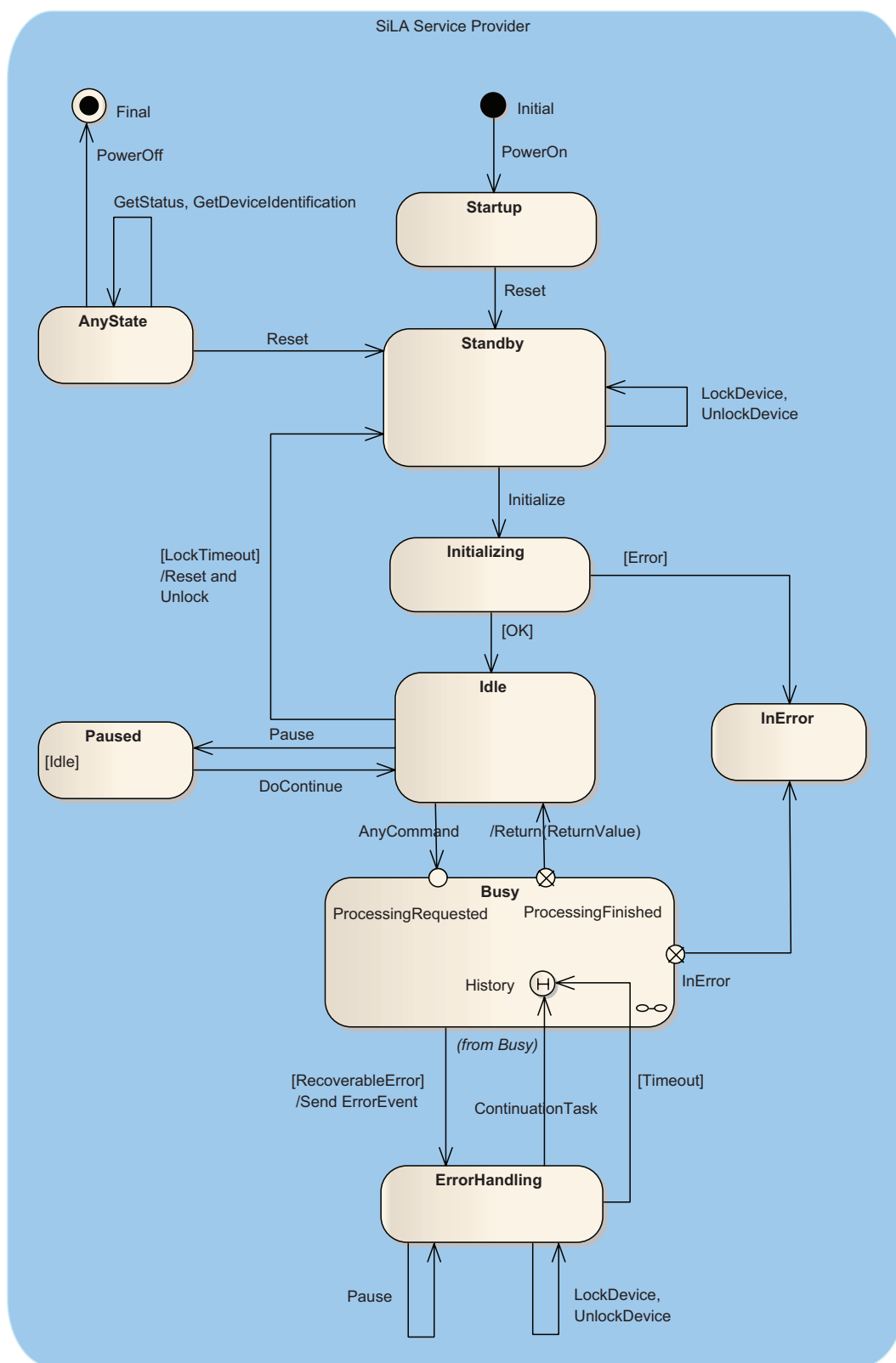


Figure 3. Main state machine. SiLA, Standardization in Lab Automation.

Considering the landscape of available devices for the drug discovery industry, a wide variability of complexity, functionality, and cost can be found. It is a challenge to press this huge diversity into a framework of standards. SiLA therefore grouped the devices into about 30 different classes. Typical device classes are pipettors, dispensers, incubators, or robotic arms. For some devices, it can be difficult to select the appropriate device class because the device has multiple functionalities that are the principal functions from different device classes. In such cases, it is also possible to define a new device class and specify those subdevice classes describing the device's functionality. To cover also low-level integrations, some special device classes were also defined, such as motor controllers or digital and analog input/output devices. They can also be integrated using a standardized SiLA interface.

From another point of view, the devices can also be classified into various complexity levels. Pipettors with multi-channel liquid handlers, robotic arms to transport plates, and additional functions are considerably more complex than a barcode reader or a simple dispenser. It was recognized early that it is impracticable to standardize all the commands that cover the complete functionality of complex instruments. Such instruments generally are delivered with sophisticated software that enables creating workflows or scripts running on them. Normally, such scripts can be named and/or stored as methods or files on the computers controlling the devices. For these complex instruments, SiLA only attempts to standardize calling the methods developed with the standard software of the device. For the simpler devices, SiLA-defined commands can cover a larger portion of the available functionality. Such collections of commands belonging to a device class are called common command sets (CCS).

The CCS of a specific device class does not cover every functionality of every device. It rather represents a minimal set of commands that should be realized and implemented for all devices in a certain device class. This condition could still be too stringent for some devices and device classes. Therefore, SiLA split the commands of the CCS into different categories. The mandatory commands are needed to maintain the state machine and therefore cannot be omitted in any SiLA interface implementation. From the CCS, two categories exist: the required commands and the optional commands. The required commands must be implemented by every device from the device class. If this is not possible, the device does not fit into the specific device class. The optional commands must be implemented only if the corresponding functionality is available. If the device has additional functionality that fits to a command definition from another device class, then the interface developer is requested to use the corresponding command for the implementation. If the additional functionality is unique or new,

then the programmer is free to implement a specific command. Such specific commands must also follow the rules that are the basis for all other commands. The command name should be descriptive for the functionality of the command. The parameter names should also be understandable and are limited to the standard data types. Specific commands should be implemented cautiously because they limit the replaceability of the devices.

In contrast to most of the commonly used programming languages, Web services or remote procedure calls do not allow having optional parameters. Therefore, the number of parameters is strictly defined in SiLA. To alleviate this limitation somewhat, selected parameters are allowed to be assigned the "empty value" (e.g., *null* in C++). The property that a parameter is "nullable" is an integral part of the definition of the command and its parameters.

Currently, SiLA has defined about a hundred commands. For each command, a definition sheet is available that contains information about the command name, its functionality, and parameters. In addition, a sample WSDL documentation tag is included, which describes the parameters and the expected return values if applicable. This template can be used by the developers for their implementations.

The collection of the commands is condensed into the SiLA Common Command Dictionary. It gives information for each command, which device class it belongs to, and whether it is a required command or an optional command.

The number of defined commands is surprisingly low considering that SiLA identified about 30 device classes. The main reason for this is that a lot of functionality is covered by commands such as *ExecuteMethod* and *SetParameter*. *ExecuteMethod* enables one to call named stored methods or scripts. *SetParameter* permits the setting of optional parameters that specialize the standard functions in a device-specific manner. The simple structure of the XML-based parameter of the *SetParameter* command enables one to set device-specific properties as name-value pairs. It is thus possible to start an action on a device with a simple command such as *Dispense* with only a minimal number of parameters but with variable presets. This approach was the only practical possibility to obtain a simple command set that ensures enough flexibility.

A few other commands are widely used by almost all device classes, such as *PrepareForInput* and *PrepareForOutput*. These commands are required to enable a transfer system like a robot, to move labware items such as plates or tubes between the devices in an integrated system.

Data Capture

Integrated automation systems typically include devices that perform sample preparation and liquid handling but also measurement devices, which measure some properties

of the samples. Most commonly, such a device is a microtiter plate reader that measures an optical, electrical, or another physical property of the sample in a single container or in a multitude of containers or wells represented as a microtiter plate. The measurements generate a stream of data, typically numerical data, that is often represented in string format for better readability. The workgroup investigated the output of more than 50 of such readers and came up with the result that it will be difficult to define a simple output format that covers all the measurement parameters from different technologies, such as fluorescence, absorption, or luminescence.

The data recorded in these files not only report the measured data but in most cases also contain meta-data that comprise information about the instrument, the identification of the labware containing the measured samples such as the barcode, and the conditions under which the measurement was performed. It is even more difficult to find a common set of descriptors that enables the reporting of such features. Therefore, the specification defines only a minimal set of required metadata items.

In addition, the specification addresses the numbering of containers in multiwell plates. Since the standard⁹ for microtiter plates of the Society for Biomolecular Sciences does not specify a numbering of wells within the plates, SiLA defined its own continuous numbering, starting with 1 at the top left corner of the microtiter plate and then incrementing the position to the right, followed by the next row also from the left to the right. This numbering scheme also enables one to describe the positions in higher integrated plates with more than 26 rows (single-letter descriptors) and containers that do not have the common rectangular shape like the compact disk-based lab disks or other specialized microfluidic devices.

In contrast to HL7, SiLA prefers an XML approach following other specifications. By this, the data structure makes use of flexibility, higher readability, and type safety. The drawback of XML is that the data volume is larger than with binary or pure text formats. Consequently, the SiLA standard allows transmitting and storing the data in a specified compressed format. SiLA provides an XML scheme that is sufficient for validation, is relatively simple, and does not contain unused overhead. So unlike the AnIML approach, which requires the use of two separate XML schemas ("AnIML Core Scheme" and "AnIML Technique Definition"), it does not take long getting used to the SiLA approach. Also, the validation of data against the SiLA schema is much easier than the validation process within AnIML.

The format of data represented by the SiLA schema is not restricted to microtiter plate-based reader data as long as the data can be efficiently represented as a string or number of a sample located in a container.

The scheme defines the representation of the following items:

- Result data
- Device meta-data
- User meta-data
- Labware information
- Plate map

The result data are stored as a set of "result series." Every result series contains the measured data, the measurement time, and information about possible exceptions during the measurement. The representation of this information is done by defining XML schema attributes. The count and data type of result series are not restricted, so the device can report multiple distinctive measurements of a multitude of sample containers as well as kinetic measurements thereof.

The device meta-data define a set of important device properties for interpretation, reproducibility, and traceability of the measurement. This comprises description of software and hardware items. Some of these properties are defined by XML schema attributes. In addition, a set of properties may not be defined by the scheme. It is designed in a fashion so that meta-data can also be included as extensions. This information, however, will not be validated against the SiLA schema. The meta-data may also comprise data related to the initiator of the measurement, such as the PMS, the user of the PMS, information about the assay, or the compounds and the labware used. For labware items such as plates and tubes, SiLA intends to provide a database with relevant geometrical and material information that can be accessed over the Internet. The meta-data included for used labware also link into this database.

Information about the role of the samples, such as "high control" or "sample" in the assay, is included in a section called the "plate map." The roles are defined as enumeration.

The data delivered by the device will be created according to the SiLA schema. Subsequently, the data representation can be validated by simple XML validation. All important information items are defined as "requested" items and will generate an error if absent.

At present, the SiLA Data Capture Specification is close to completion. It focuses on data delivered by microtiter plate readers. Because of the huge diversity of meta-data from available readers, the goal is not to define all possible meta-data.

In addition to the structure of the data, their handling between the devices, the PMS, and high-level data processing and storing entities (DPE) have to be considered. The first possibility is to store the measured data in the device. Only a link to these data will then be transferred from the device to the PMS, which also transfers only these links to the DPE. In the second variant, the data themselves

(without images) are transferred to the PMS. The PMS stores the data and delivers the data to the DPE. It is planned to allow both variants in the standard.

The workgroup is now working on fine-tuning the SiLA schema, which should be in review by the SiLA members early in 2012.

Realization Status of the Standards

The integration of laboratory automation systems based on the SiLA standards requires the availability of SiLA-compliant devices and PMS that can control the devices. Because there is one standard protocol between the devices and the PMS, the integration of devices into a system is much easier than in the past. In addition, the format of the data delivered by measurement devices is compliant to a SiLA standard. Therefore, the PMS and subsequent analysis software must cope only with this single format, instead of one proprietary format for every measurement device.

The establishment of the SiLA standards is currently in a transition phase where appliers would like to integrate existing non-SiLA-compatible devices and also be prepared for the integration of newly developed devices with the native SiLA interface. SiLA supports such integrations by allowing different integration levels that comprise software-only device driver interfaces as well as intelligent hardware interface converter-based interfaces.

To support such developments, the *Infoteam SiLA Library* from Infoteam Software (Stäfa, Switzerland) describes a smooth path to offer a SiLA interface either as a protocol converter box—whether installed on a separate computer or on the PMS computer—or as a native component of a device driver. The *Infoteam SiLA Library* implements all common aspects of the mandatory commands and the state machine of the SiLA standard. The device manufacturer only has to add device-specific parts. This saves a lot of time and avoids implementation failures.

To integrate devices of different vendors into a laboratory automation system, integrators created large collections of drivers for different devices. The *Infoteam SiLA Library* offers a simple and fast way to integrate SiLA-compliant devices into a PMS.

For the setup of an integrated laboratory automation system based on SiLA-compliant devices, a PMS also is needed. Such software is available, for example, from EQUIcon Software GmbH (Jena, Germany). Its product, *niceLAB*, is designed to allow the definition and execution of simple or complex parallel methods of collaborating SiLA devices. The workflow editor is fully graphic oriented and allows a very easy definition of the method and its parameters. There is also a plug-and-play mechanism for the integration of new SiLA devices within a few minutes into the system based on the interpretation of the WSDL documentation tag. No development activities are necessary to do this. With this

functionality of *niceLAB*, a user can build his or her own laboratory automation system without assistance from an integrator. This new approach is especially designed for the construction of small and flexible systems. The applier only has to get the necessary SiLA-compliant devices, do the mechanical integration, connect it for communication, and configure the PMS in a simple manner. In this way, the user can save time and money.

A first application of *niceLAB* was the built from a SiLA-compliant system of three devices in a pharma company. Because of the features explained above, this was possible to do within a few hours.

Xavo AG (Reinach, Switzerland) created a product called *Lab Logistics* based on the SiLA standards. It can be used to manage the use and location of microtiter plates and compounds. The system acts similar to a PMS; it not only controls any plate movements but also keeps its logistics database up to date.

Lab Services (Breda, the Netherlands) made its existing PMS *Plate Butler* SiLA compliant. It allows the scheduling of plate movements between different medical sample preparation and component analyzing systems. If a SiLA-compliant device driver is available, a user can easily integrate a new device into the system.

The Hamilton Company (Bonaduz, Switzerland) developed a SiLA converter for its Decapper to use it as SiLA-compliant device. Additional devices are under development. The company also adapted its PMS, used to control the flexible liquid-handling systems, so that it could integrate SiLA devices. All these SiLA-compatible equipment components and software were shown at the Lab Automation trade show in January 2011. This setup proves that SiLA devices can be integrated in automation processes. By using the SiLA interface standard, Hamilton is able to integrate its own devices and also devices of other manufacturers into custom systems in a rapid way.

Different members of the SiLA consortium have developed drivers to prepare devices with SiLA compliancy. Applications have been written that make use of SiLA devices. With the PMS described above, a small system has been implemented and tested.

One pharma member company plans to equip a screening laboratory with SiLA devices and integrated systems complying with the SiLA standards. For this setup, Thermo Scientific (Langensfeld, Germany) will launch a new product—a composition of several storage units that can be controlled as a single SiLA device. The SiLA approach allows accessing the composite device by managing PMS alongside another component that is used for environmental monitoring.

First instruments with a native SiLA interface have been announced recently by different companies at the 2011 Association for Laboratory Automation conference in Palm Springs, California, and more are expected to follow soon.

Visit the SiLA Web site at www.sila.coop for more details and further examples. The SiLA organization will update the Web site regularly with more success stories.

Conclusions

The SiLA organization has defined three basic standards for the rapid integration of laboratory automation systems performing various tasks in drug discovery research and other lab automation applications. With these standards and additional optional training offered from the SiLA organization, firmware developers can implement SiLA-compliant devices, and system integrators can use SiLA devices in their systems. Although the number of finished, started, and planned implementations is still limited, these efforts clearly show some of the benefits for integrators and appliers. For such pilot implementations, SiLA-compatible drivers for a number of devices had to be programmed. SiLA drivers can be set up within reasonable time and cost frames, if they are based on generic frameworks that implement the SiLA state machine and the SiLA mandatory commands. The experience of the authors with such implementations illustrates that simple integrations of three to five devices into SiLA-compliant PMS software can be achieved in a matter of a few hours. For example, a test system consisting of a plate transport robot, a dispenser, and an incubator was integrated in less than a day and could be controlled by a prototype PMS software. The fast integration was possible because the devices were able to communicate their capabilities and command set through the WSDL. This could then be interpreted by the PMS and fed into its graphical user interface. Specialized application programming is also easily possible using the SiLA interface definitions and the SiLA device-specific CCS, as exemplified in two small systems combining a number of different barcode and dot code readers for the registration and anonymization of clinical samples in the laboratories at F. Hoffmann–La Roche. To benefit even more from the standard, the availability of SiLA-compatible devices is key. The first of such devices is currently appearing on the market.

The decision to implement a system based on a standard is influenced by many factors that mostly represent a short-term view of the situation, including availability of instruments, cost of instruments, estimated integration time, and the status and reputation of the putative standard. Factors such as maintenance cost, longevity of devices, availability of support, and maintenance over the long term are often only considered secondary.

It is clear to most people involved in laboratory automation that it would be nice and useful if SiLA interfaces would already have been implemented and SiLA-compliant devices would be widely available. However, in reality, it will be a long way until all required elements, such as

devices and PMS software, are available and interact with each other in the intended ways. Therefore, many players in the field are reluctant to invest in a new technology, even though it might be superior, make it easier to replace devices, facilitate long-term maintenance, simplify the firmware development, or enable the user to better understand the inner workings of the system. It is often easier to stick to solutions that have worked in the past and only to gradually try to optimize these and not to bet on a new development that has a chance of not being successful.

The SiLA organization is well aware of these problems and therefore has tried to keep the entry hurdles at a minimum by allowing different practicable ways to implement the SiLA interface for legacy devices so that these instruments can be integrated into SiLA-compliant laboratory automation systems and PMS software easily. Support for the implementation of such solutions is also offered by specialized members of the consortium.

Outlook

The implementation of integrated laboratory automation systems using SiLA-compatible equipment has clear advantages for suppliers, integrators, and appliers. Suppliers benefit from the standard by streamlining their firmware and software development. Integrators profit from a unified interface of the devices, which removes the huge variability in hardware and software that has to be dealt with. This will make the PMS software more independent of the devices and eliminates the need for device driver development. The modern Web service-based architecture of the SiLA interface opens up new fields of applications that were not possible before, such as access to devices from different controlling software and remote monitoring of devices in automated laboratory systems. Appliers benefit from increased flexibility and modularity of their integrated systems and more sustainable devices that are independent of computer operating systems and PMS software, thus promising longer lifetimes as well as better reusability. It is also expected that the applier will have more choices to select an appropriate PMS that fits the needs of his or her task. Furthermore, it will be simpler to understand the behavior of the instruments, which in turn will help to widen the scope of laboratory automation within the research organizations.

At the current stage of development of the SiLA standard, all involved parties such as device manufacturers, system integrators, and system appliers are confronted with extra investment both in cost and labor. With the spread of the standard, this should be reduced significantly in the near future when more devices and software will be available and when firmware developers can rely on a stable standard.

The SiLA organization works hard to foster relationships between device suppliers, system integrators, and

laboratory automation appliers in the bio-pharma industry. Although the initial germination of the SiLA idea was in Switzerland and has extended significantly into Europe, it was always the goal to reach out to the major players in the United States and globally. The SiLA standard can only be successful if it is adopted industry-wide. Such an adaptation cannot be immediate but will require considerable time and up-front investment. It is essential that all involved parties invest money, resources, and time with the prospect to gain in the future and that they do not wait until one of the others has progressed. It would especially be beneficial if a considerable number of US device suppliers, integrators, and bio-pharma companies could join the effort. In particular, the formation of an American center of SiLA excellence would be desired. From there, new members could be acquired and new standard development initiatives could be started in accordance with the European center.

It is understandable that supplier and integrator companies that have well-established products in the marketplace are reluctant to change them and invest heavily in a standard development effort. It is certainly easier to join if one is at the beginning of an innovation cycle. But it should be considered that in our fast-changing world, even innovative products can be outdated very quickly. Actively contributing to the further development of a standard such as SiLA is certain to protect current and future investments.

Declaration of Conflicting Interests

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The authors received no financial support for the research, authorship, and/or publication of this article.

References

1. Hawker, C. D.; Schlank, M. R. Development of Standards for Laboratory Automation. *Clin. Chem.* **2000**, *46*, 746–750.
2. American Society for Testing and Material. Standard Specification for Transferring Information between Clinical Instruments and Computer Systems. In *Annual Book of ASTM Standards*, vol. 14.01; American Society for Testing and Material: West Conshohocken, PA, 1991.
3. Smith, B.; Ceuters, W. *HL7 RIM: An Incoherent Standard. Study in Health Technology and Informatics*; IOS Press: Fairfax, VA, 2006.
4. Bray, T.; Paoli, J.; Sperberg-McQueen, C. M.; Maler, E.; Yergeau, F. Extensible Markup Language (XML) 1.0 (Fifth Edition) [Online]. 2008. <http://www.w3.org/TR/REC-xml>
5. Object Management Group, Inc. Laboratory Equipment Control Interface Specification [Online]. 2002. <http://www.lecis.org/documents/CORBA%20LECEIS%2002-09-13.pdf>
6. Roth, A.; Jopp, R.; Schäfer, R.; Kramer, G. W. (2006). Automated Generation of AnIML Documents by Analytical Instruments. *J. Lab. Autom.* **2006**, *11*, 247–253.
7. Box, D.; Ehnebuske, D.; Kakivaya, G.; Layman, A.; Mendelsohn, N.; Nielsen, H. F.; Thatte, S.; Winer, D. Simple Object Access Protocol (SOAP) 1.1. [Online]. 2000. <http://www.w3.org/TR/2000/NOTE-SOAP-20000508>
8. Christensen, E.; Curbera, F.; Meredith, G.; Weerawarana, S. Web Services Description Language (WSDL) 1.1 [Online]. 2001. <http://www.w3.org/TR/wsdl>
9. ANSI/SBS 1-2004; ANSI/SBS 2-2004; ANSI/SBS 3-2004; ANSI/SBS 4-2004. <http://www.slas.org>