TySSA – A Set of Means for Building of Distributed Software Systems for the Automation of Experiments by the User

Shvetsov V.N., Salamatin K.M., Salamatin I.M, Tsulaia M.I.

FLNP, JINR, Joliot Currie 6, Dubna, Moscow region, Russia

Modification of software systems for automation of experiments requires time comparable to their development due to uniqueness. In this regard, methods and means that reduce these terms are relevant. The paper presents a set of tools designed to build distributed software systems for automation of experiments (SAE). Service-oriented architecture is used. The main means of the complex are multi-user programs. Each SAE, built by means of the complex **TySSA**, consists of independent components (programs in the format .exe). If the necessary services are available, their integration into the experiment automation system corresponding to the planned experiment technique is performed by the experimenter and requires 10-20 minutes. The complex **TySSA** is designed for use in experiments at the **IBR-2** research reactor [1], but can be used in other problem areas.

1. Architecture of the complex

The emergence of large neutron centers leads to the expansion of the research front and requires a quick response of developers of research automation systems. The possibilities of modern computer technology allowed to create a technology for the development and modification of software systems for the automation of experiments that meets this requirement.

The developed complex **TySSA** (Typical System for Spectrometry Automation) uses Service-Oriented Architecture (SOA), which provides effective approach to solving the problem.

1.1. Basic principles of the used architecture

1. **Publication of access interfaces**. Services are components of the automation system, they publish their interfaces and interact with each other and with support services through open, widely used standards. Interfaces do not depend on the platform, programming language, operating system and other technical features of the implementation.
2. **Separation of code**. The service included in the information system implements a separate function, which is a logically separate, repetitive task used in accordance with the experimental technique.
3. **Large-Block structure of services**. Services in SOA are components of a sufficiently high level, so that the interaction between them is reduced to a limited number of messages.
4. **Weak coupling**. Services can be implemented independently of other system services. Changes to the service implementation do not affect the control program that uses the service. Weak coupling provides a simple adaptation of the system to changes in the structure and composition of services.

These principles allowed to remove the most difficult problems of integration of components in SAE and to react to changes in a configuration and a technique dynamically and without difficult transformations at the level of integration of components. The architecture is aimed at supporting a process rather than a program. The components are integrated in such a way that their execution takes place in a certain sequence in accordance with the technique of the experiment. Each of the service components can participate in different automation systems.

1.2. Technological support of architecture

The following main components have been developed to support SOA technology:

1. **DiCME** module [2] (message bus), which provides guaranteed sending and receiving of messages.
2. **Database** (DB) - an electronic catalog where information about each component of the system is stored. The database management program provides customers with information about the components (services) that are currently available.
3. **Program for description of technique of work**.
4. **Experiment control program** - a component that controls the flow of work in accordance with a given technique. At the same time, data processing at separate stages is carried out in different independent components.
5. **Services** to ensure the implementation of the technique. The set of selected services, the specified sequence and parameters of their operation determine the technique of the experiment.
6. **Additional group of services**, whose task is to monitor emergency situations, ensure the viability of the system, etc.

2. PSJ support database management program

The database is implemented by means of **SQLite** and in 4 tables contains information about the characteristics of available services (**devices** and **SrvsProps**), experiment management programs **(JOBS**, **CurrentJob**) of all users and some current information. Figure 1 shows the database management interface and the contents of the devices table. The figure shows the names and ways of presenting data to control the operation of services. The data to update the devices table are provided by the developers of these programs.

There are 2 modes of filling the database: (1) editing by using the provided (figure 1) interface; (2) automatic, by reading the passport of the service in case of it change. Information from the devices table is used to describe the technique of the experiment and to prepare the experiment program.

3. PSJ program for description of the technique and preparation of the experiment program in JSON format

The PSJ program is designed to describe the technique of the experiment and prepare the text of the task for the experiment (experiment program) in JSON format. The technique is represented by a sequence of states of the experimental setup, in each state data registration is performed. The state of the instrument is formed by the sample environment devices, the composition of the devices and the parameters transmitted to them are determined by the

experimental technique. Each device is managed by a separate service. The number of devices and states is not limited.

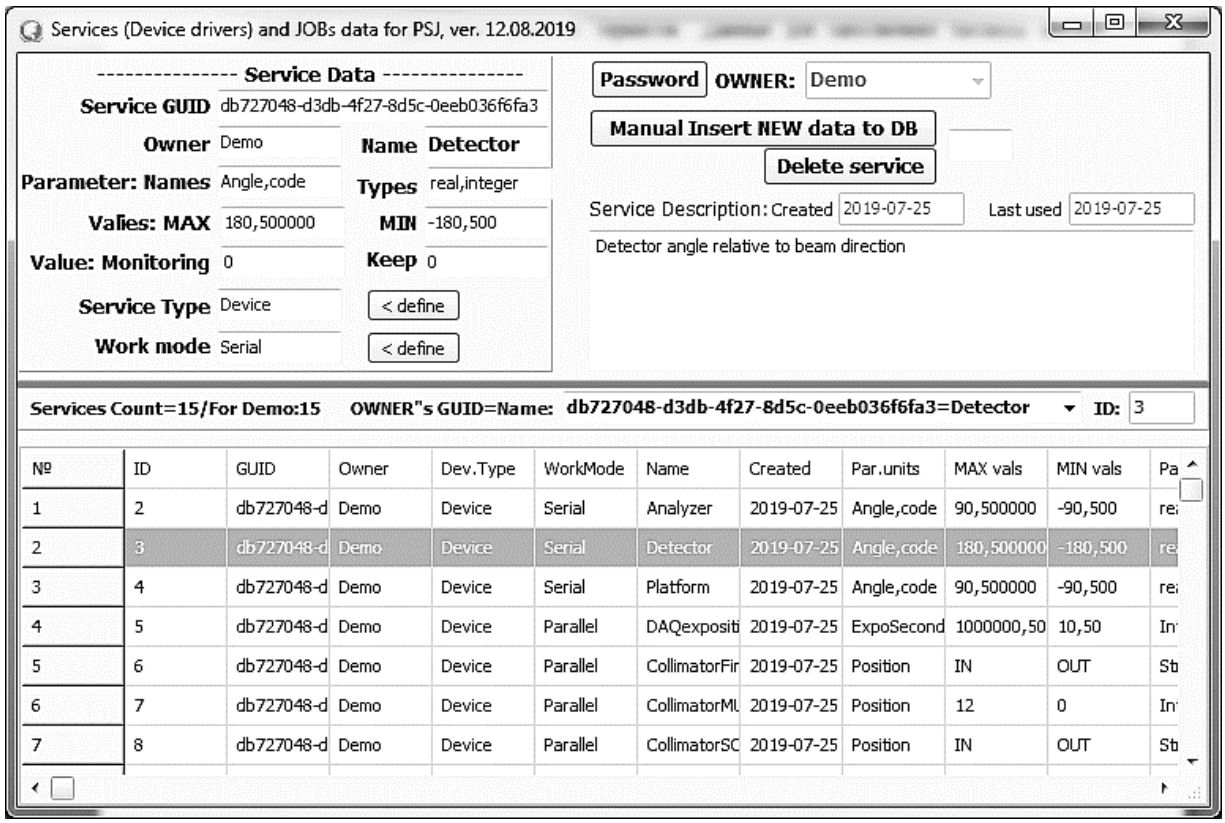

Figure 1. PSJ support database management program interface.

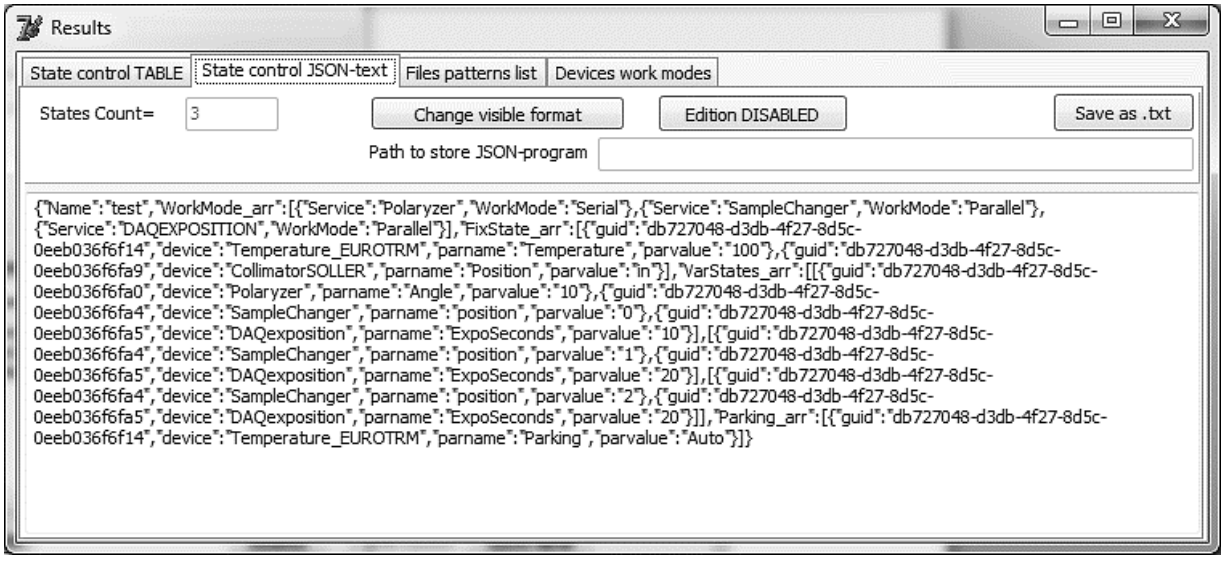

Figure 2. Text of the experiment program.

The PSJ program allows you to select the characteristics of the necessary services from the database in accordance with the planned technique of the experiment and make a task for the experiment.

The experiment program contains 3 main sections:

1. A section of fixed conditions that remain unchanged throughout the experiment.
2. A section of changeable conditions. This section describes all the instrument states in which you want to register data, as well as the parameters for recording and storing data. The number of implemented states and registration conditions are determined by the planned experiment technique.
3. A section providing parking of devices at the end of work.

Figure 2 shows the text of the experiment program prepared by the PSJ program. The PSJ program enters the results and intermediate data of the task preparation process into the database.

4. Job Control program

The **Job Control** program performs the task for the experiment that was prepared using PSJ. Search for services in the network and interaction with them is performed by a specially designed module **DiCME**. The **Job Control** program is equipped with means of protection against loss of information in case of power failures and failures of experimental equipment.

5. Services that manage sample environment devices

Within the framework of the considered technology, a special structure of services and a method of integration with the experiment management program were developed. Services are independent components in executable format (.exe) and are combined into a distributed system dynamically.

Each service manages one sample environment device. The set of devices determines a number of states of the experimental setup at which the registration of experimental data will be performed.

The structure of the service can be divided into the following parts:

1. A device **Passport** describing some program IDs, managed parameters and ranges of their values.
2. The **main function** of the service, forming the conditions on the device in accordance with the parameters transmitted to it.
3. A function that returns the **time required** to perform this operation.
4. The **Interface** part that provides interaction with the control program.

Figure 3 shows a fragment of the service program text containing an example of the Passport view. Different formats for parameters representing can be used.

The interface part of the service provides management of the service over the network, which includes:

1. The discovery of a service and determining of address for the interaction.
2. Transfer of control parameters to the service.
3. Receiving service responses about the duration of the operation completion.
4. Receiving service responses on completion of the operation.

```
{Passport Data}
const
  SRVC_NAME      = 'Polarizer';
  SRVC_GUID      = 'db727048-d3db-4f27-8d5c-0eeb036f6fa0';
  SRVC_TYPE      = 'device';
  SRVC_OWNER     = 'Demo';
  SRVC_WORK_MODE = 'serial';
  SRVC_PAR_NAMES = 'angle,code';
  SRVC_PAR_TYPES = 'real,integer';
  SRVC_PAR_MIN   = '-90,500';
  SRVC_PAR_MAX   = '90,10000';
  SRVC_HAVEMONITORING = false;
  SRVC_HAVEKEEPVALUE  = false;
```

Figure 3. Fragment of program text of service.

The text of the interface part is the same for all services in this technology. For the new service, only the text shown in figure 3 (Passport) and two functions are edited and combined with the interface part. The interaction of the experiment control program (**Job Control**) with the service is performed by the **DiCME** module, using open protocols **SLP, JSON-RPC, MQTT** [3]. These protocols, using **GUID**, provide search of the necessary service in a local network and management of its work.

All services that put the hardware system in a needed state can run simultaneously. The hardware part of the system may include several computers connected to a local network. In this case, the service is started on the corresponding computer and the control program searches for it automatically.

The same devices (e.g. collimators, filters, targets, DAQ controllers, etc.) can be used in different installations. If the software is created using this technology, it is possible to use the same services. To ensure that the characteristics of a particular instance of the service are found in DB the service passport contains the name of the installation (**Owner** field).

6. Example of DAQ subsystem parameterization for TOF-experiments

The **DAQ** subsystem differs from other services in the way it interacts with the experiment control program. The description of time encoder used in a number of experiments is presented in paper [4]. For this device, at the stage of describing the experimental technique, only the duration of exposition in each state of the experimental setup is determined. The command to start the registration is issued by **Job Control** automatically, when each installation of state is completed, the registration is stopped automatically by the **DAQ** subsystem when the specified exposition time expires. The operator commands **Pause, Continue** do not affect the actual exposition duration. Parameterization of the method of operation of the time encoder requires much more parameters, the values of which are determined by the requirements of the selected experimental technique are transmitted to the **DAQ** subsystem in a file.

Conclusion

A multi-user complex of tools for building distributed SAE with service-oriented architecture was developed.

The proposed mechanism of interaction of system components according to protocols implemented in **DiCME** and **SLP** allows different users in the same local network to use unified programs, provides freedom to expand the functionality and configuration of a distributed software control system. Using dual task-specific service identity (**GUID** and **Owner** codes) eliminates conflicts when using services with the same name and code in different applications on the same LAN.

The proposed structure of the service simplifies their development, ensures the continuity of these components of the system and gives a significant saving of SAE modification time when changing the experimental technique. In addition, this approach encourages the standardization of a number of sample environment devices.

Using a unified experiment management program and services in executable format (.exe), the ability to use ready-made services in different systems, dynamic integration of components into the software system of the experiment provide a significant reduction in the time of development or modification of software systems for specific experiments. The same way of saving data makes it possible to use ready-made statistical control utilities, visualization, data processing in different experiments.

The proposed technology for developing software systems is more effective in large neutron centers, where dozens of neutron guides are used, the equipment used (devices for changing collimators, filters, targets, **DAQ** systems, etc.) is standardized and often there is a need to change the experimental technique.

References

1. https://elibrary.com.ua/m/articles/view/PULSING NUCLEAR REACTOR IBR-2M.
2. *Salamatin K.M.* DiCME - distributed environment for interaction of components of the experimental automation system for low-energy physics // Software engineering, 2014. No. 3, p.p. 3–11.
3. Service Location Protocol RFC 2608.
4. *Shvetsov V.N.; Alpatov S.V.; Astakhova N.V.* et al. An eight-channel system for neutron-nuclear investigations by the time-of-flight method // Instruments and Experimental Techniques, Vol. **55**, No. 5, pp. 561–568, 2012.