

# EPICS PV Management and Method for RIBF Control System



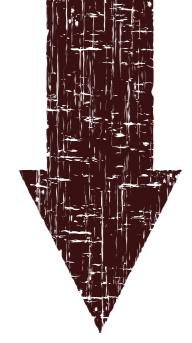
◎ Akito Uchiyama#, Misaki Komiyama, Nobuhisa Fukunishi (RIKEN Nishina Center)

## Abstract

For the RIKEN Radioactive Isotope Beam Factory (RIBF) project, the Experimental Physics and Industrial Control System (EPICS)-based distributed control system is utilized in Linux and vxWorks. Utilizing network attached storage (which has a high-availability system) as a shared storage, common EPICS programs (Base, Db, and so on) are shared by each EPICS Input/Output Controller (IOC). From the initial development of RIBF control system, it has continued to grow and consisted of approximately 50 EPICS IOCs and more than 100,000 EPICS records. Because RIBF has been constructed by extending RIKEN Accelerator Research Facility (RARF) in a previous project, the controllers for RARF are also utilized for RIBF control system. In this case, the dependence between the EPICS records and EPICS IOCs becomes complicated. For example, it is not easy to know the accurate EPICS record name information using only the device information. Therefore, we constructed a new management system for the RIBF control system to easily call up the detailed information. In the system, by parsing startup script files (st.cmd) to run EPICS IOCs, all EPICS records and EPICS fields are stored in the PostgreSQL-based database. By utilizing these stored data, we succeeded in developing Web-based management and search tools.

## Motivation

For system maintenance and development of Channel Access client, we need to identify the IOC hostname connected to the PVs.



- RIBF control system is constructed by extending RARF control system (previous project).
- Not use UDP broadcast for ca\_search (EPICS\_CA\_AUTO\_ADDR\_LIST=NO)
- We would like to provide efficient system environment to search IOC hostname from PV name easily for developers and operators.

## Development of PV management system for EPICS-based control system (Similar concept of IRMIS@ANL[1])

### Method of System Construction

By reading the startup script files and accessing the EPICS runtime database files, the program can parse the runtime database files. Therefore, we developed a program such that the information is separately stored in the PostgreSQL-based database by parsing the file.

- EPICS Substitution file and macro are also available.

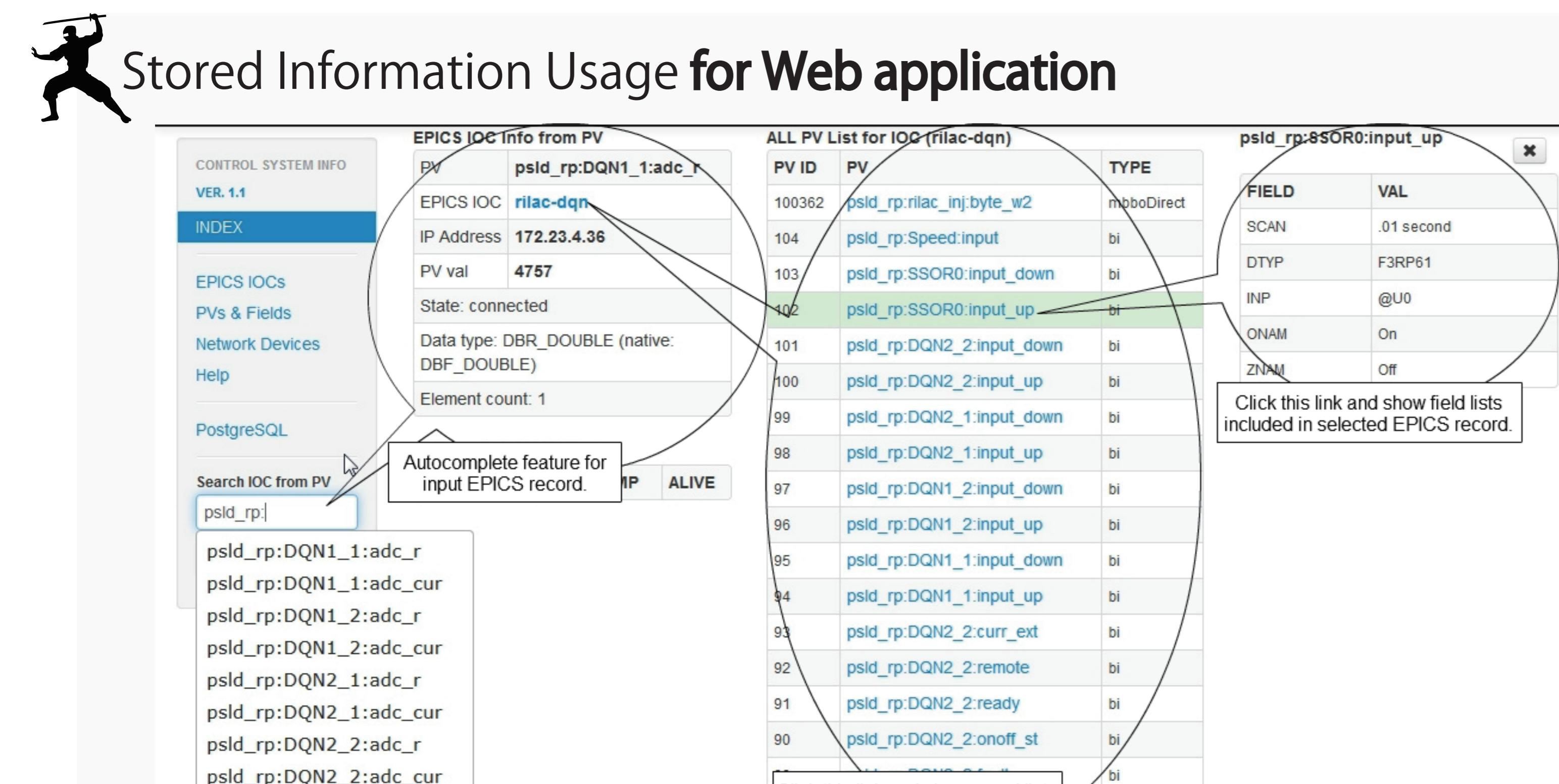
```
Startup script file
#!/bin/sh
# You may have to change example to something else
# everywhere it appears in this file
<envPaths
ca:$TOP>
# Register all support components
dbLoadTemplate "dbExample.db"
example_registerRecordDeviceDriver pdbbase
# Load record instances
dbLoadTemplate "dbUserHost.substitutions"
dbLoadRecords "db/dbExample.db", "user=rootHost"
drvAsynIPPortConfigure("netwk1", "172.23.3.221:999", 0, 0, 1)
db/dbExample.db
recordGo,$(user)cathodeCurrentC {
    field(DESC,"Set cathode current")
    field(PV,"Soft Channel")
    field(SCAN,"1 second")
    field(DRVC,"5")
    field(PREC,"2")
    field(EGU,"Amps")
    field(DRVN,"20")
    field(OPR,"10")
    field(HPR,"20")
    field(OPR,"0")
}
}

Database
fieldinfo
```

Table Name	Information Stored in the Column	Number of Records
iocinfo	Hostname of IOC	51
	Directory path of EPICS application	
	Startup script file name	
pvinfo	EPICS record name	110,192
	Record type	
fieldinfo	Field	3,151,383
	Field type	
device2ioc	Hostname for network-based device	432

Table : Database Structure Used for Management System in the RIBF Control System (October 2015).

### Stored Information Usage for Web application



The screenshot shows a web application interface for managing EPICS PVs. On the left, there's a sidebar with 'CONTROL SYSTEM INFO' and a search bar for 'psld\_rp:'. The main area displays three tables: 1) 'EPICS IOC Info from PV' showing a single result for 'psld\_rp:DQN1\_1adc\_r'. 2) 'ALL PV List for IOC (Filac-dqn)' listing multiple PVs like 'psld\_rp:ilac\_inj\_byte\_w2', 'psld\_rp:Speed\_input', etc. 3) 'psld\_rp:850R0:input\_up' showing detailed fields for a specific record. A tooltip indicates an 'Autocomplete feature for input EPICS record'.

- We can search EPICS IOC hostname from EPICS PV name. (⇒ cainfo)
- Without requiring the completed EPICS record name, because of autocomplete feature.
- We can check all records and fields without source code.

### RIBF Control System and Shared Storage

Common EPICS programs (EPICS-base, application programs, runtime database, and additional extensions programs) are stored in the NAS, and they are shared by all EPICS IOCs using the NFS or FTP.

IOC Platform	Connected Control Device	Type	Number of IOCs
Linux x86	• N-DIM • PLC • GPIB • Other network-based devices	Soft IOC	24
Linux x86	• CAMAC	Embedded IOC	6
Linux f3RP61	• PLC	Embedded IOC	14
vxWorks	• NIO	Embedded IOC	7

Diagram illustrating the shared storage architecture. A central 'NAS (NetApp)' unit provides NFS (read-only) access to a 'Management Server' and a 'Linux IOC'. It also provides NFS (read-write) access to a 'Development Server (For Linux IOCs)' and an 'vxWorks IOC'. Additionally, it provides FTP access to another 'Development Server (For vxWorks IOCs)'.

Table : Current Status of the Type of EPICS IOC

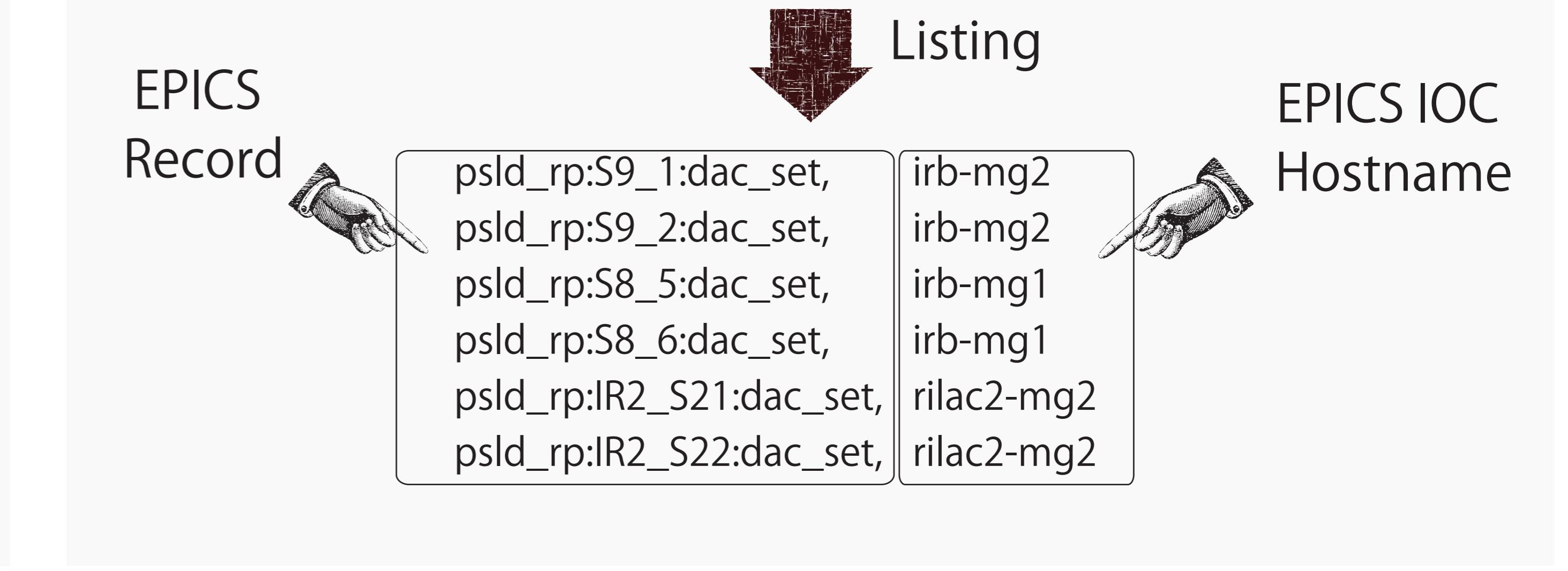
### Stored Information Usage for Command-line Tool

## It is very easy to make PV lists by a program !!

• We use this feature to make PV list for caMonitor and the electric logbook.  
• Some command-line tools are developed to obtain the information from DB.

For example, making a list for all of the PVs including "psld\_rp:XXXX:dac\_set".

\$sql1="SELECT \* from \$table where PV like  
psld\_rp :%:dac\_set and active='1' order by pv\_id";

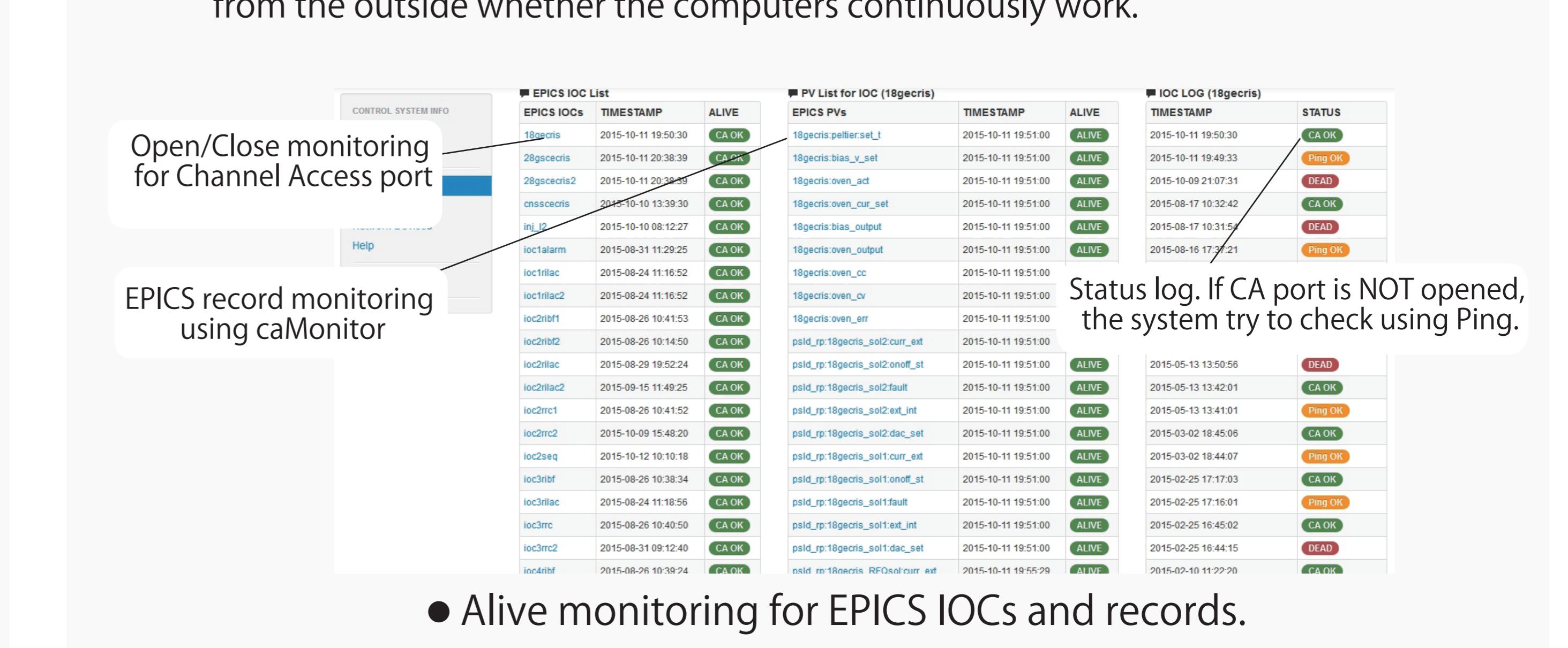


The diagram shows a hand pointing from a box labeled 'EPICS Record' to a box labeled 'EPICS IOC Hostname'. Below them is a list of PV names: psld\_rp:S9\_1:dac\_set, psld\_rp:S9\_2:dac\_set, psld\_rp:S8\_5:dac\_set, psld\_rp:S8\_6:dac\_set, psld\_rp:IR2\_S21:dac\_set, psld\_rp:IR2\_S22:dac\_set, irb-mg2, irb-mg2, irb-mg1, irb-mg1, rilac2-mg2, rilac2-mg2. An arrow points down to this list from the text '\$sql1="SELECT \* from \$table where PV like psld\_rp :%:dac\_set and active='1' order by pv\_id";'

### Stored Information Usage for Alive Monitoring

(90% completed)

Generally, alive monitoring is performed by a system that checks from the outside whether the computers continuously work.



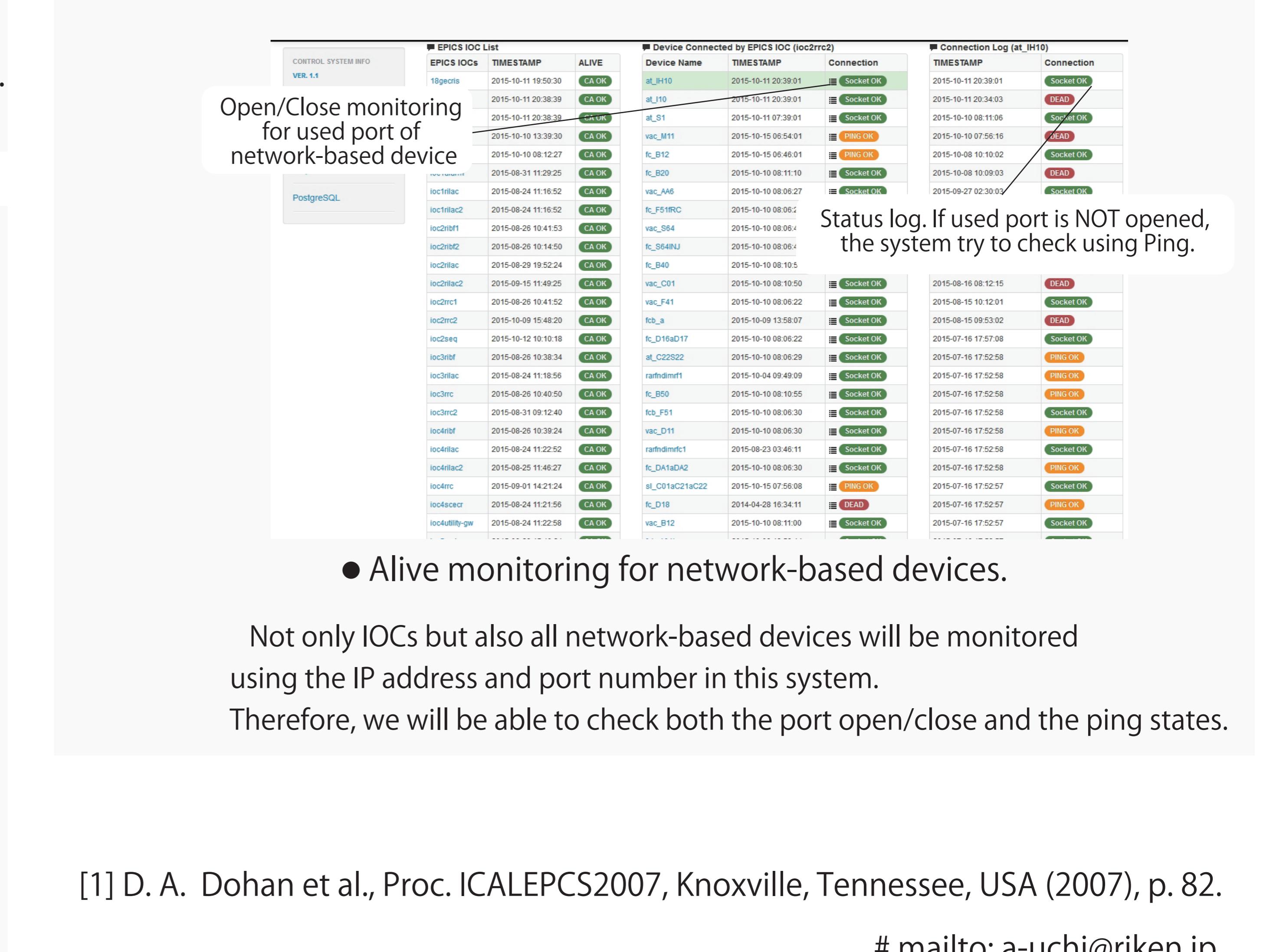
The screenshot shows a web application for alive monitoring. It includes sections for 'Open/Close monitoring for Channel Access port', 'EPICS record monitoring using caMonitor', and 'Status log'. The status log table shows entries for various IOC and record names with their status (e.g., ALIVE, DEAD, UNKNOWN).

- Alive monitoring for EPICS IOCs and records.

### Stored Information Usage for Alive Monitoring

(90% completed)

Generally, alive monitoring is performed by a system that checks from the outside whether the computers continuously work.



The screenshot shows a web application for alive monitoring of network-based devices. It includes sections for 'Open/Close monitoring for used port of network-based device', 'Device Connected by EPICS IOC (epc2rrz)', and 'Connection Log at (epc2rrz)'. The connection log table shows entries for various device names with their connection status (e.g., Socket OK, READ, WRITE, UNKNOWN).

- Alive monitoring for network-based devices.

Not only IOCs but also all network-based devices will be monitored using the IP address and port number in this system.

Therefore, we will be able to check both the port open/close and the ping states.

[1] D. A. Dohan et al., Proc. ICALEPCS2007, Knoxville, Tennessee, USA (2007), p. 82.

# mailto: a-uchi@riken.jp