WLOADCTL

Enterprise Workload Automation Platform

WLOADCTL Product Documentation – Installation & Deployment Manual

Enterprise-Grade Workload Control
Scheduling Engine & Distributed Architecture

Version 1.0

Document Description

Document Positioning

This document is intended for WLOADCTL deployment, operations, and implementation personnel. It provides guidance for completing installation, initialization, functional verification, and product registration of the platform under different deployment scales.

Scope and Applicable Versions

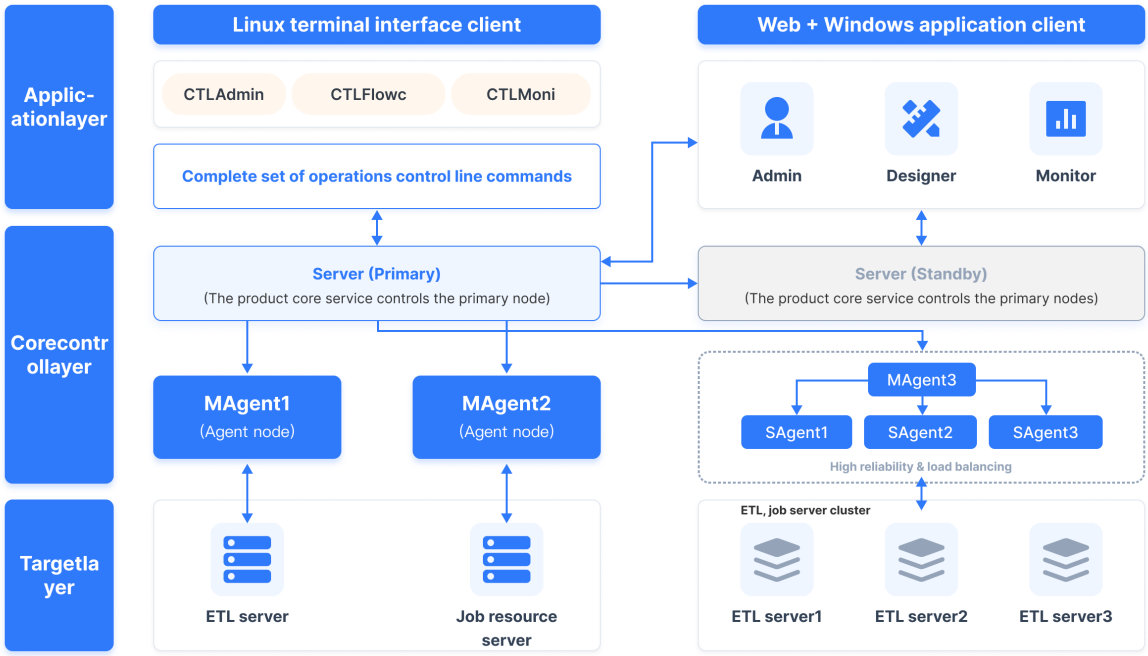
- Applicable Product: WLOADCTL
- Applicable Scenarios: Standalone Deployment / Distributed Deployment / High Availability Deployment
- Applicable Version: wloadctl 8.0.046

- Applicable Environments: Test Environment / Pre-production Environment / Production Environment

Product and Deployment Overview

Product Architecture Overview

The WLOADCTL automation scheduling standard product adopts a typical **B/S + C/S hybrid architecture**:



- Application Layer: Client layer, including Admin / Designer / Monitor
- Control Layer: Server-side scheduling and control core
- Target Layer: Scheduled business systems (such as ETL servers, job workstations, etc.)

The platform adopts a multi-level control architecture, with inter-layer communication based on the TCP protocol:

- The control layer follows a multi-tier pyramid architecture
- The top layer is the scheduling service control node
- The agent layer achieves cluster scheduling and load balancing through master-slave cascading

Core Component List

- WLOADCTL Server (Scheduling Control Core), Mandatory
- WLOADCTL Agent (Execution Agent), Optional
- WLOADCTL Console Client (Character-Based Client), Mandatory
- Web Application Server (B/S Application Server), Optional
- Web Browser (B/S Application Client), Mandatory
- WLOADCTL DB Store (Database), Optional

Deployment Modes and Delivery Architecture

Overview of Deployment and Delivery Modes

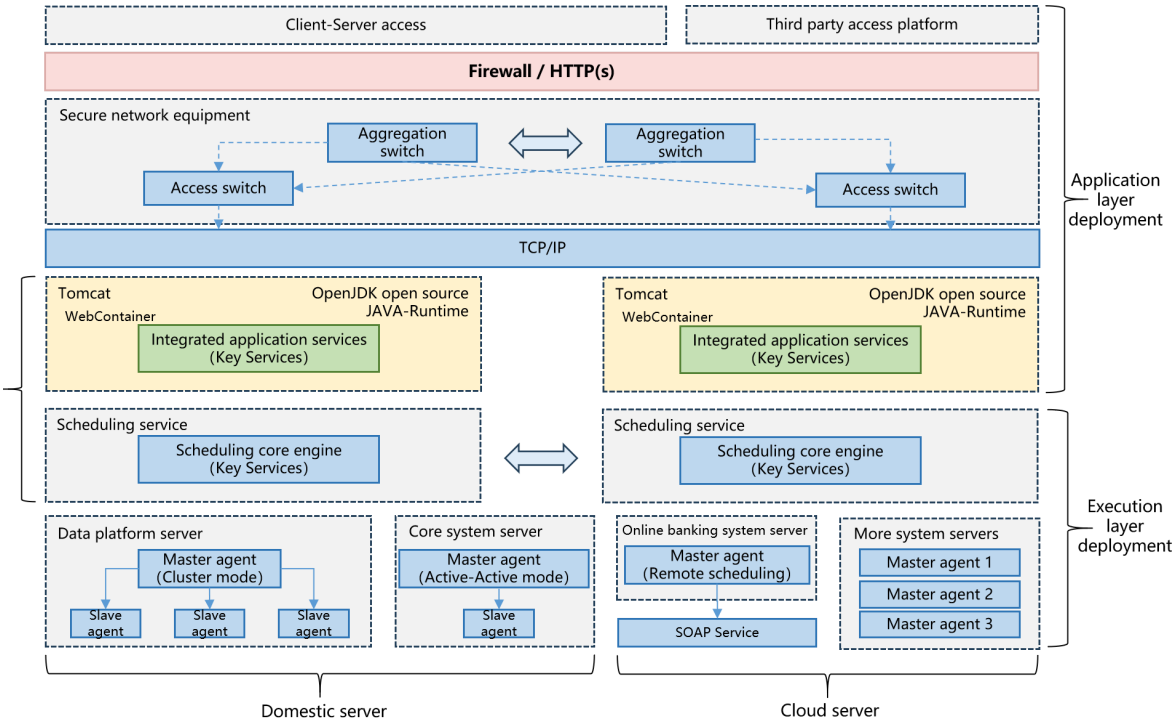
WLOADCTL is typically deployed as an **infrastructure-level scheduling platform** in data centers or private cloud environments. The deployment must meet the following requirements:

- Long-term stable operation
- Elastic scalability with business growth
- Multi-operating system compatibility
- Cross-network and cross-region scheduling capability
- Convenient operation, maintenance, and upgrade mechanisms

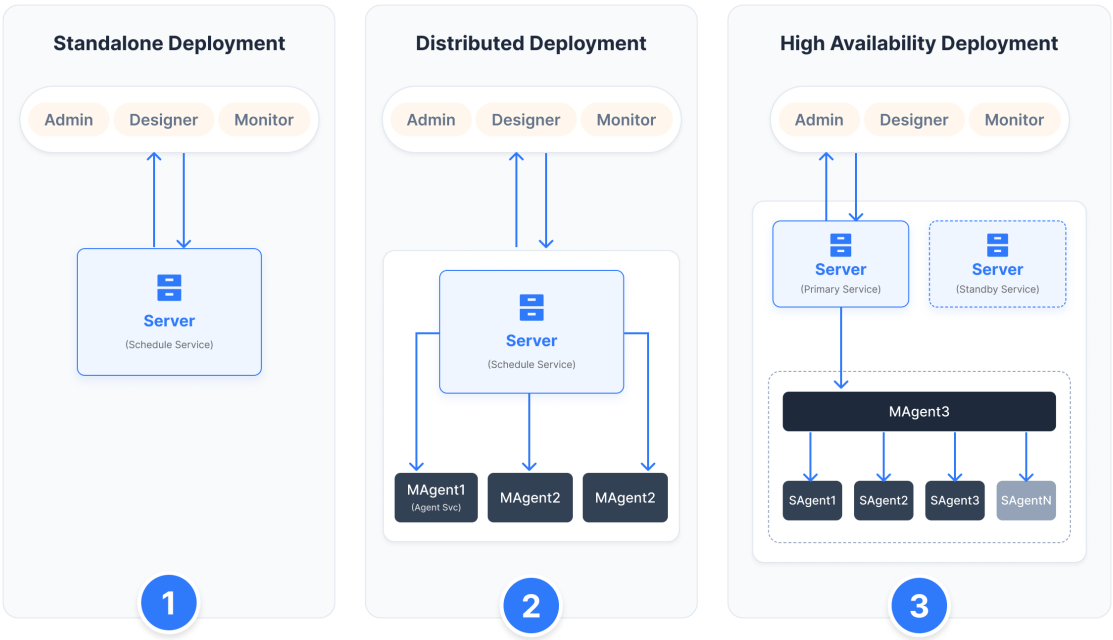
Network and Delivery Architecture

The platform supports the following typical network topologies:

- Single data center deployment
- Multi-business-zone deployment
- Cross-network-region deployment
- Cross-system security domain deployment



Based on unified deployment requirements and network architecture, WLOADCTL provides three mainstream deployment solutions for enterprises, supporting smooth evolution from standalone to distributed and eventually to high-availability architecture.



Deployment Solution Description

Standalone Deployment Solution

Objective Positioning

The standalone deployment solution is primarily intended for **initial validation** and **project-level application** scenarios of enterprise scheduling requirements.

- **Rapid Validation**: Quickly implement scheduling capabilities, validate system feasibility, and establish a technical foundation for subsequent large-scale deployment.
- **Small-Scale Scenarios**: Suitable for independent batch processing requirements of a single project or single system, supporting task volumes in the thousands and focusing on efficient single-point business execution.

Architectural Advantages

This model adopts a lightweight design with relatively low deployment and maintenance costs:

- **Easy Deployment, Low Cost**: Simple architecture, short deployment cycle, minimal personnel and hardware investment, and low maintenance cost.
- **Independent Security**: Scheduling services of each system operate independently without interference; core business systems and management systems can be deployed separately to avoid fault propagation, providing high security and stability.

Architectural Limitations

There are significant limitations in unified global management and cross-system coordination:

- **Lack of Unified Governance**: Systems are deployed independently, making unified global monitoring impossible. Operations personnel must log in to multiple nodes, resulting in lower efficiency.
- **Weak Collaborative Scheduling Capability**: Cross-system business dependencies cannot be managed internally through scheduling. External event triggering is required, making it

difficult to precisely control the overall batch processing time window and achieve fine-grained cross-system scheduling optimization.

Distributed Deployment Solution

Objective Positioning

The distributed deployment solution is primarily positioned to address scalability and management challenges in **enterprise-level medium to large-scale business environments**.

- **Scalable Expansion**: Adapts to continuous business growth, meeting the requirements of **high-concurrency scheduling** and **large-scale batch processing**.
- **Unified Management**: Establishes enterprise-level unified scheduling standards (scheduling specifications, monitoring views, message alerts, and operations management), supporting task volumes in the tens of thousands and hundreds of agent nodes, enabling centralized global system governance.
- **Complex Business Adaptability**: Supports multi-system scheduling management and task orchestration in complex business chain scenarios.

Architectural Advantages

This architecture fully leverages distributed technology characteristics and provides the following advantages:

- **Strong Horizontal Scalability**: Core components (scheduling control, orchestration management, execution agents, and operations management) support clustered deployment and can achieve horizontal expansion by adding nodes.
- **Unified Scheduling and Operations Platform**: Through unified scheduling technology, monitoring views, and alert mechanisms, enterprise-level scheduling standards are established. Global version updates, upgrades, start/stop operations, and other tasks are simplified, improving operations efficiency.
- **High Reliability**: Tasks are distributed across multiple agent nodes for execution, avoiding single points of failure. The cluster architecture supports fault tolerance and load balancing to meet production environment stability requirements.

Architectural Limitations

There is a certain level of complexity in implementation and operations:

- **High Implementation Complexity**: Requires unified planning of overall technical standards and architectural design, involving multi-service integration and coordination. The implementation cycle is longer and demands higher technical team capabilities.
- **Heavier Management and Maintenance Burden**: The management complexity of distributed clusters (network communication, node health checks, load balancing, etc.) is higher than that of a monolithic architecture.
- **High Dependency on Hardware and Network**: Requires a stable network environment and sufficient hardware resources. Network fluctuations or hardware failures may directly affect scheduling accuracy and execution stability.

High Availability Deployment Solution

Objective Positioning

The **high availability deployment model** is primarily positioned to address **business continuity** and **extreme fault tolerance** requirements.

- **Ultimate Reliability**: Suitable for **mission-critical business scenarios** (such as core financial systems) with extremely high stability requirements. Eliminates single points of failure and ensures continuous availability of the scheduling platform.
- **Disaster Recovery and Backup**: Provides internal system fault tolerance and **cross-regional disaster recovery** capabilities (such as two sites and three data centers), capable of handling physical-level disasters.
- **Large Scale and High Performance**: Supports horizontal expansion and vertical performance enhancement, suitable for enterprise-level unified scheduling platforms with task volumes in the hundreds of thousands and thousands of agent nodes.

Architectural Advantages

This architecture is centered around **redundancy design** and **rapid recovery**, offering the following advantages:

- **No Single Point of Failure**: Scheduling control, orchestration management, and operations management services adopt **hot standby deployment**. When the primary node fails, the standby node automatically takes over to ensure continuous platform operation.
- **Rapid Fault Tolerance Capability**: Execution agents are deployed in multi-node clusters. When a single node encounters an exception, tasks can quickly switch to healthy nodes for execution.
- **Disaster Recovery Capability**: Supports a two-site, three-data-center disaster recovery mechanism. If one regional center fails, other regional centers can take over the business.
- **Flexible Switching**: Load balancing enables traffic distribution and fault isolation, supporting **online service switching** and **rapid recovery** of failed nodes.

Architectural Limitations

There are higher requirements in terms of resource investment and management complexity:

- **High Resource Consumption**: To achieve multi-node redundancy and disaster recovery, server resources and network bandwidth investment must be significantly increased.
- **Higher Overall Cost**: Both hardware and maintenance costs increase substantially. Additional resources are required for node management and maintenance.
- **Long Implementation Cycle**: Involves multi-node networking, load balancing configuration, disaster recovery data synchronization, and other tasks, typically requiring several weeks to complete planning and implementation.

Environment and Preconditions

Pre-deployment Checklist

Before installing WLOADCTL, **the following checks must be completed**:

Operating System and Permission Check

- A dedicated runtime user has been created (it is not recommended to run directly as root)
- The runtime user has read, write, and execute permissions for the installation directory

[Example commands, for reference only]

```
su - root
useradd -m wloadctl
su - wloadctl
```

System Time and Time Zone Check

- All Server / Agent nodes have completed time synchronization (NTP)
- All nodes maintain consistent time zones

[Example commands, for reference only]

```
timedatectl
chronyc tracking
```

Network and Port Check

- The scheduling service port is not occupied
- Bidirectional network connectivity between Server and Agent
- One-way reachability from the application server to the Server
- One-way reachability from the scheduling server to the database service
- One-way reachability from the application server to the database service
- One-way reachability from the client to the application server

[Example commands, for reference only]

```
ss -lntp | grep 18581
ping <agent-host>
```

Dependency and Resource Check

- CPU, memory, and disk resources meet the minimum configuration requirements
- The database service is started and can be connected normally

[Example commands, for reference only]

```
free -h
df -h
```

Operating System and Hardware Requirements

Standalone Deployment

Purpose	Operating System	Memory	CPU	Disk	Quantity	Remarks
Scheduling Service Node	Linux 5.4+ 64-bit	16G	8C	200G	1	
Web Application Server	Linux 5.4+ 64-bit	8G	4C	200G	1	
Web Client	Desktop OS	8G	4C	100G	n	Use modern browsers

Distributed Deployment

Purpose	Operating System	Memory	CPU	Disk	Quantity	Remarks
Scheduling Service Node (Primary)	Linux 5.4+ 64-bit	16G	16C	500G	1	Standby machine is physically separated from the primary machine
Scheduling Service Node (Standby)	Linux 5.4+ 64-bit	16G	16C	500G	1	Standby machine is physically separated from the primary machine
Web Application Server (Primary)	Linux 5.4+ 64-bit	16G	8C	500G	1	Can reuse the primary server environment
Web Application Server (Standby)	Linux 5.4+ 64-bit	16G	8C	500G	1	Can reuse the standby server environment
Job Execution Node	Linux 5.4+ 64-bit	—	—	—	n	Network connectivity with primary/standby servers, can reuse existing business hosts
Database Service Node	Linux 5.4+ 64-bit	16G	8C	1000G	2	Supports advanced data analysis functions (optional)
Client	Desktop OS	16G	8C	100G	n	Use modern browsers

High Availability Deployment

Purpose	Operating System	Memory	CPU	Disk	Quantity	Remarks
Scheduling Service Node (Primary)	Linux 5.4+ 64-bit	16G	16C	500G	1	Standby machine is physically separated from the primary machine
Scheduling Service Node (Standby)	Linux 5.4+ 64-bit	16G	16C	500G	1	Standby machine is physically separated from the primary machine
Web Application Server (Primary)	Linux 5.4+ 64-bit	16G	8C	500G	1	Provides JDK 17, with reusable primary server environment.
Web Application Server (Standby)	Linux 5.4+ 64-bit	16G	8C	500G	1	Provides JDK 17, with reusable primary server environment.
Job Execution Node	Linux 5.4+ 64-bit	—	—	—	n	Network connectivity with primary/standby servers, can reuse existing business hosts
Database Service Node	Linux 5.4+ 64-bit	16G	8C	1000G	2	Supports advanced data analysis functions (optional)
Web Client	Desktop OS	16G	8C	100G	n	Use modern browsers

Network and Port Planning

- Scheduling service port (example: 18581)
- Web service port (example: 8088)
- Proxy communication port (example: 18589)

Software and System-level Preconditions

- It is recommended to configure the system message queue of the scheduling server as follows:

```
------ Messages Limits --------
# Thousands of jobs scale
default max size of queue (bytes) >= 1638400
# Tens of thousands of jobs scale
default max size of queue (bytes) >= 16384000
# Hundreds of thousands of jobs scale
default max size of queue (bytes) >= 163840000
```

- JDK version required for Web application service: 17
- Database type: MySQL 8, character set UTF-8

Installation Package Description and Acquisition Method

Obtaining the Installation Package

- Official download address: www.wloadctl.com
- Please download the installation package matching the CPU architecture of the target server
- Deployment packages for special UNIX environments such as AIX and HP-UX need to be obtained by contacting the official team

Description of Installation Package Composition

- Backend core system deployment package (including server, agent, and character interface client)
- Web application (B/S client) deployment package

Installation Package Directory Structure

Directory Structure of Scheduling Server and Agent

Directory	Description
bin	Directory for storing running programs
conf	Directory for storing basic configuration files
flowcfg/debuglog	Directory for temporary running logs of jobs debugged in Designer
flowcfg/cout	Directory for storing released versions of container info in Designer
flowcfg/enterp	Directory for storing saved versions of container info in Designer
flowcfg/compiled	Directory for storing compiled versions of container info in Designer
flowcfg/pubhis	Directory for storing archived versions of container info in Designer
lib	Directory for storing secondary development and system-dependent library files
log	Directory for storing system-level logs
src	Directory for storing plug-in development and system info access sample programs
tmp	Directory for storing temporary data files during system operation
upgrade *	Directory for storing version upgrade-related information
commfile	Directory for storing job execution communication files
tmpjoblog	Directory for storing temporary log files of job execution
history/tmp *	Directory for storing temporary files of business chains
history/daydata *	Directory for storing historical records of job execution
history/jobdone	Directory for storing OK files of job execution (for donejob usage)
history/res *	Directory for storing historical resource info of scheduling nodes
history/runlog *	Directory for archiving scheduling logs
history/busschain *	Directory for storing business chain information
history/dispathnum *	Directory for storing daily running job counts of nodes
history/dayreport *	Directory for archiving daily scheduling reports
history/daylog *	Directory for archiving job logs
history/um_appreaded *	Directory for storing user-read messages
history/flowlog *	Directory for storing container historical records

Directory	Description
sync/source *	Static files to be synchronized for active-standby synchronization
sync/target *	Identification files for active-standby synchronization
sync/spedata *	Dynamic files to be synchronized for active-standby synchronization
work/code	Directory for storing container info in Monitor workspace
work/other	Directory for storing other info in workspace
work/log	Directory for storing current logs of containers in Monitor workspace
work/data	Directory for storing container data in Monitor workspace
work/event	Directory for storing event status
work/forknum	Directory for storing active-standby synchronization info
work/ipcs	Directory for storing IPC resource info of nodes
work/kernel	Directory for writing back runtime Kernel info of nodes
work/usermsg	Directory for storing user-subscribed messages
work/anlyset	Directory for storing initialization data of business chains
work/syncno	Directory for storing active-standby synchronization info
work/svrimport	Directory for ctlsvrimport import logs
work/jdinfo	Directory for storing job node information
work/calendar	Directory for storing scheduling calendars
include	Directory for storing secondary development header files
licence	Directory for storing authorization files
spool	Cache directory for non-scheduling component programs
ctlvm	Actual storage directory for installation files (other directories are soft links to this)
doc	Directory for storing help documents
demo	Directory for process control files (XML) and binary development program examples

Application Server Directory Structure

Directory	Description
config	SpringBoot configuration files (e.g., application.yml)
dist	Directory for packaged products of front-end projects
lib	Directory for storing third-party dependent library files
log	Directory for system running logs

Standard Installation Process

Installation Process Overview

Based on the product's core architecture and installation package structure, product installation is divided into two major stages: backend core system installation and client application system installation.

1. **Backend core system installation**: Includes scheduling service deployment and scheduling agent deployment.
2. **Client application system installation**: Includes character interface client deployment and graphical interface application deployment.

The character interface application has the same environment requirements as the core system, so it is integrated into the same installation package (backend software installation package) to simplify the installation process.

The product's core architecture consists of scheduling service nodes and scheduling agent nodes, so the core system installation is divided into two independent stages:

- Each platform **must deploy exactly one** scheduling service node.
- Agent nodes are flexibly deployed according to business scale and deployment environment.
- In standalone mode, the service node has a built-in agent node, so only the service node needs to be installed.

The core system adopts a multi-layer network architecture and is sensitive to IP and port management. Administrators need to plan and allocate them in advance.

Core Scheduling Service Node Installation

Adjust System Core Parameters

1. Log in as the root user, edit the `/etc/sysctl.conf` file, append the following configurations and save:

```
kernel.msgmax=81920
kernel.msgmnb=16384000        # Refer to "Software and System-level
Preconditions" for specific values
kernel.shmmax=18446744073692774399  # It is recommended not to exceed 60% of
physical memory
```

2. Execute the following command to make the configurations take effect:

```
sysctl -p
```

```
[root@node-144-EN ~]# sysctl -p
vm.overcommit_memory = 1
kernel.msgmni = 32000
kernel.msgmax = 81920
kernel.msgmnb = 163840000
kernel.shmmax = 18446744073692774399
[root@node-144-EN ~]# ipcs -l

------ Messages Limits --------
max queues system wide = 32000
max size of message (bytes) = 81920
default max size of queue (bytes) = 163840000

------ Shared Memory Limits --------
max number of segments = 4096
max seg size (kbytes) = 18014398509465599
max total shared memory (kbytes) = 18446744073709551612
min seg size (bytes) = 1

------ Semaphore Limits --------
max number of arrays = 32000
max semaphores per array = 32000
max semaphores system wide = 1024000000
max ops per semop call = 50
semaphore max value = 32767
```

Prepare the Environment

1. Execute the following command to create a dedicated user:

```
useradd -m wloadctl
```

2. Configure environment variables: Edit `.bash_profile` in the installation user's directory and add the following configurations:

```
export TASKCTLDIR=$HOME/wloadctl      # Product installation directory
export PATH=$PATH:$TASKCTLDIR/bin:.  # Add to execution path
```

3. Make the environment variables take effect:

```
source .bash_profile
echo $TASKCTLDIR
```

Installation Steps

1. Log in as the user and extract the installation package:

```
su - wloadctl
tar -zxvf cir_linux_amd64_8.0.xxx_xxxxxxxx.tar.gz
```

2. Execute the installation program:

```
cd cirinst
./inst
```

3. Enter the character installation interface:

```
WLOADCTL Job Automation Technology Platform Installation Software.
Version info: linux(64) 8.0.046
------------------------------------------------------------
Installation Notes:

Set these environment variables for the current user before installing.
Environment variables:
    1.export TASKCTLDIR=[Product Installation Path]
    2.export PATH=$PATH:$TASKCTLDIR/bin:.
For example:
    export TASKCTLDIR=$HOME/taskctl
    export PATH=$PATH:$TASKCTLDIR/bin:.
------------------------------------------------------------
```

4. After pressing Enter, the installation program automatically checks the environment variables. If the check passes, the following is displayed:

```
WLOADCTL Job Automation Technology Platform Installation Software.
version info: linux(64) 8.0.046
------------------------------------------------------------

        1. Classic Installation    2. Custom Installation
        3. Uninstall              4. Exit

Option Descriptions:
    1. Classic Installation: Suitable for most scenarios.
    Primarily used for single-machine scheduling control, managing only one
CtlNode.
    2. Custom Installation: for cross-platform multi-machine collaboration,
    high-availability services, and distributed scheduling.
------------------------------------------------------------------
Official Website: www.wloadctl.com
Copyright:  WLOADCTL Co., Ltd.

  Please select:2

Confirm? (y/n)y
```

5. Select **2 Custom Installation** to enter the node type selection interface:

```
WLOADCTL Job Automation Technology Platform Installation Software Custom
Installation.
-------------------------------------------------------------

        1. Agent node Installation    2. Character Interface Client Installation
        3. Server node (HA Primary)  4. Server node (HA Standby)
        5. Server node (No HA)
        6. return
-------------------------------------------------------------
Copyright:  WLOADCTL Co., Ltd.
Official Website: www.wloadctl.com
  Please select:5

Confirm? (y/n)y
```

6. Select **5 Server node (No HA)** and enter the IP and port of the current service node as prompted:

```
-------------------------------------------------------------
-                         Server node Installation.          -
-------------------------------------------------------------

Start unpack ...
Unpack complete.

Start copying files to product directory...
Copy files to product directory completed.

Start cleaning up temporary installation files on Server node...
Server node temporary installation file cleanup completed.

Enter installation configuration information:
-------------------------------------------------------------
Input the IP address of the current Server node. [Default is 10.40.0.144]:
Input the port of the current Server node. [Default is 18581]: 48581

Your current input as follows:
-------------------------------------------------------------

  Current Server node IP                  :10.40.0.144
  Current Server node PORT                :48581
-------------------------------------------------------------

Confirm? (y/n)y
```

- IP address: Valid IP address of the local machine
- Port: Unoccupied scheduling service port

7. After confirming the information, the system automatically completes the installation:

```
-------------------------------------------------------------
-              Character interface client installation.          -
-------------------------------------------------------------

Start unpack ... ...
Unpack complete
```

```
Start copying files to product directory.... ...
Copy files to product directory completed.

Start cleaning up temporary installation files on client...
Client temporary installation file cleanup completed.

Version '8.0.046' use success

~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Notice ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~                                                                  ~
~    Be sure to remember the 'ctlhelp' command,                     ~
~    which can help you learn how to start and stop CtlNode,        ~
~    and other background operation commands.                       ~
~                                                                  ~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Install successful.
```

8. Execute the `ctlinit` command to initialize the scheduling service node:

```
Server node begins initialization....
The listening port of the Server is: 48581

Initialization successful. Current released version: 8.0.046, compiled version:
Feb  3 2026 15:49:16
```

9. After installation is complete, you can view node information through `ctlninfo 1`:

```
ctlninfo 1

linux@wloadctl CtlNode 'svrnode' basic information:
-------------------------------------------------------------------------
version        : linux(64) 8.0.046     | id             : 2
type           : 2-Server node         | desc           : Server Node
state          : 2-Stopped             | statedesc      :
ip(domain)     : 10.40.0.144           | port           : 48581
isactive       : Activated             | hasHA          : No HA
svrbegintime   :                       | hisdispatchnum : 0
acttime        :                       | currdispatchnu : 0
searchinteval  : 0                     | fdcsleep       : 0
-------------------------------------------------------------------------
runmaxnum      : 100                   | currrunnum     : 0
currffnum      : 0                     | maxvirresval   : 1000
currvirresval  : 0                     | currffvirres   : 0
cpu_user       : 3.70                  | memtotal       : 15.36
memused        : 63.95                 | memthreshold   : 100.00
diskuse        : 47.00                 | diskthreshold  : 97.00
-------------------------------------------------------------------------
```

The high-availability version uses the same installation package and requires corresponding authorization. During deployment, select **3 Server node (HA Primary)** for the primary node and **4 Server node (HA Standby)** for the standby node. The remaining steps are similar, but you need to additionally enter the peer node IP, port, and automatic takeover policy.

Core Scheduling Agent Node Installation

The core architecture of the system consists of **scheduling service nodes** and **scheduling agent nodes**. A built-in agent node is default in standalone deployment; agent nodes must be deployed to achieve cross-machine communication in cross-server scheduling scenarios.

The service node and agent node use the **same installation package**, and the deployment type is only distinguished by installation options.

```
WLOADCTL Job Automation Technology Platform Installation Software Custom
Installation.
------------------------------------------------------------------
        1. Agent node Installation    2. Character Interface Client Installation
        3. Server node (HA Primary)  4. Server node (HA Standby)
        5. Server node (No HA)
        6. return
------------------------------------------------------------------
Copyright:  WLOADCTL Co., Ltd.
Official Website: www.wloadctl.com
  Please select:1
Confirm? (y/n)y
```

Key operation instructions:

1. After the agent node installation is complete, you must execute `ctlinit` to complete initialization to ensure the agent service runs normally.
2. In high-concurrency scenarios, it is recommended to adjust message queue and shared memory parameters with reference to the parameter configuration requirements of the service node.

Client and Web Service Installation

Practical operation shows that the Monitor module has high requirements for real-time monitoring performance (delay is recommended ≤5 seconds); the Admin and Designer modules have relatively low resource requirements, which can be met by mainstream client hardware.

Prepare the Environment

1. Create a dedicated runtime user:

```
useradd -m wloadctl
su - wloadctl
```

2. Check the Java environment (JDK 17 or above is required):

```
java -version
java version "17.0.18"
Java(TM) SE Runtime Environment (build 17.0.18+8-LTS)
```

Installation Steps

3. Upload and extract the Web installation package:

```
tar -zxvf wloadctl-service-8.2_xxxxxxxx.tar.gz
drwxrwxr-x 2 wloadctl wloadctl     150 Dec 25 11:28 config
drwxrwxrwx 4 wloadctl wloadctl      71 Jan 23 18:12 dist
-rwxrwxr-x 1 wloadctl wloadctl    6611 Dec 10 10:28 install.sh
drwxr-xr-x 2 wloadctl wloadctl    8192 Nov 26 14:03 lib
drwxr-xr-x 2 wloadctl wloadctl    4096 Feb  5 11:32 log
-rwxr-xr-x 1 wloadctl wloadctl    2231 Dec 24 11:53 service.sh
-rw-r--r-- 1 wloadctl wloadctl 1121762 Dec 30 15:20 wloadctl-web-service.jar
```

5. Enter the directory and execute the installation script, configure Web and server information as prompted:

```
sh install.sh
********************************************************************************
**********************   WLOADCTL V8.0 Online Installer   ***************
**********************        Powered by wloadctl.com       ***************
********************************************************************************
Existing WLOADCTL server installation detected. Using the current server IP and
port.
Enter WLOADCTL Server IP (default: ): 10.40.0.144
Enter WLOADCTL Server Port (default: 48581):
Enter WLOADCTL Web Application IP (default: ): 10.40.0.144
Enter WLOADCTL Web Application Port (default: 8088): 9401
********************************************************************************

WLOADCTL Server IP        : 10.40.0.144
WLOADCTL Server Port      : 48581
WLOADCTL Web App IP       : 10.40.0.144
WLOADCTL Web App Port     : 9401
********************************************************************************

Confirm installation? [Y/n]: y

Installation completed successfully!
Please run ./service.sh start
Access URL: http://10.40.0.144:9401/
```

6. Start the Web application service:

```
./service.sh start
...
15:20:21798 [main] INFO  org.apache.coyote.http1.Http11NioProtocol - Starting
ProtocolHandler ["http-nio-9401"]
15:20:21.928 [main] INFO
org.springframework.boot.web.embedded.tomcat.TomcatWebServer - Tomcat started on
port(s): 9401 (http) with context path ''
2026-02-10T15:20:22.781+0800: [GC (Allocation Failure) [PSYoungGen: 253920K-
>8160K(253952K)] 296172K->68504K(516096K), 0.0286691 secs] [Times: user=0.12
sys=0.00, real=0.03 secs]
15:20:23.389 [main] INFO  org.springframework.boot.SpringApplication - Started
application in 12.232 seconds (JVM running for 14.689)
```

7. After successful startup, access it through a browser: `http://<Application IP>:<Port>`. The product registration wizard will be displayed for the first installation; refer to the "Product Registration" chapter for the specific process.

Initialization and Configuration

Preconditions for Initialization

Before platform initialization, you need to start the listening of service nodes and agent nodes first, and confirm that the nodes are in a stopped state. The listening operation process of service nodes and agent nodes is consistent.

The `ctlinit` command is used to start node listening, with the following functions:

1. Start port listening for internal communication of the platform.
2. Apply for shared memory space for loading and storing platform configuration information.
3. Create message queues for inter-process communication.

After the listening is successfully started, you can verify it through the following commands:

- View shared memory: `ipcs -m`

```
ipcs -m
------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch     status
0x04520395 229395     cir80_en   666        77524984   80
0x04520398 229396     cir80_en   666        993973864  55
0x0452038f 229397     cir80_en   666        378340024  8
0x04520389 229398     cir80_en   666        96         5
0x045222d5 229409     wloadctl       666        77524984   10
```

- View message queues: `ipcs -q`

```
ipcs -q
------ Message Queues --------
key        msqid      owner      perms      used-bytes   messages
0x045222cb 458803     wloadctl       666        0            0
0x045222cc 458804     wloadctl       666        0            0
0x045222cd 458805     wloadctl       666        0            0
0x045222ce 458806     wloadctl       666        0            0
0x045222cf 458807     wloadctl       666        0            0
0x045222d0 458808     wloadctl       666        1536         3
0x045222dc 458809     wloadctl       666        0            0
```

The system provides the `ctlshut` command to stop node listening. For more background commands, you can query through `ctlhelp`.

Initialization Methods and Content

Platform initialization is mainly completed through the **Admin client**, which adopts graphical interface configuration with intuitive and systematic operations.

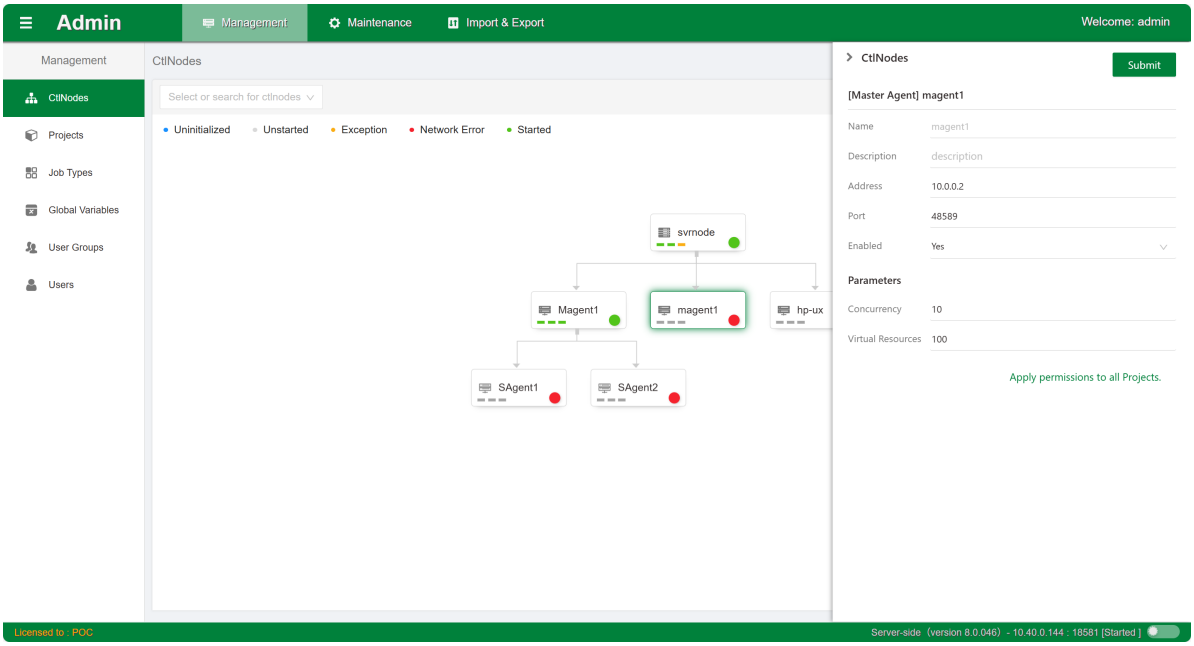
Initialization mainly includes the following key steps:

1. **Node configuration**: Define the type of each node and establish the superior-subordinate management relationship.

2. **Basic information**: Preset job types and complete the basic configuration of application projects.
3. **Information loading**: Load platform configuration information into the shared memory of service nodes and agent nodes.

Platform Node Definition

Complete node definition on the `Platform Nodes` page of the B/S client.



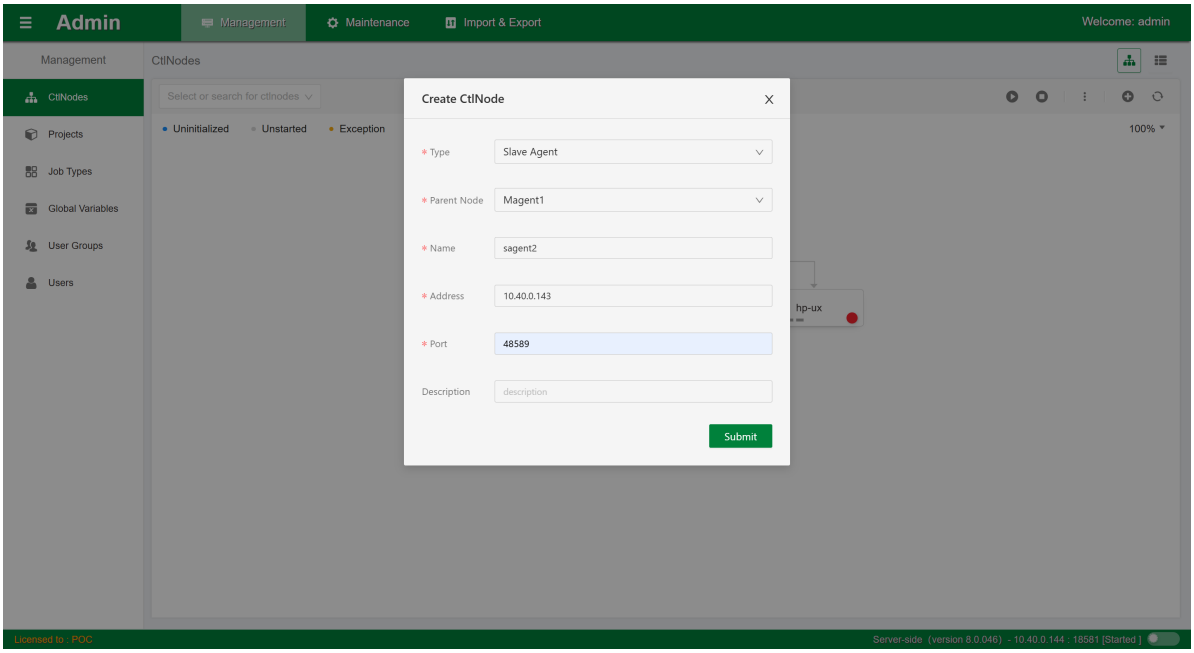
- **Define scheduling server node**

A default scheduling server node (svrnode) is automatically created when logging into the system for the first time. Only the node name and remarks can be modified, and attributes such as IP and port are preset by the system.

- **Define master agent node**

  Add a master agent node on the `Platform Nodes` page:

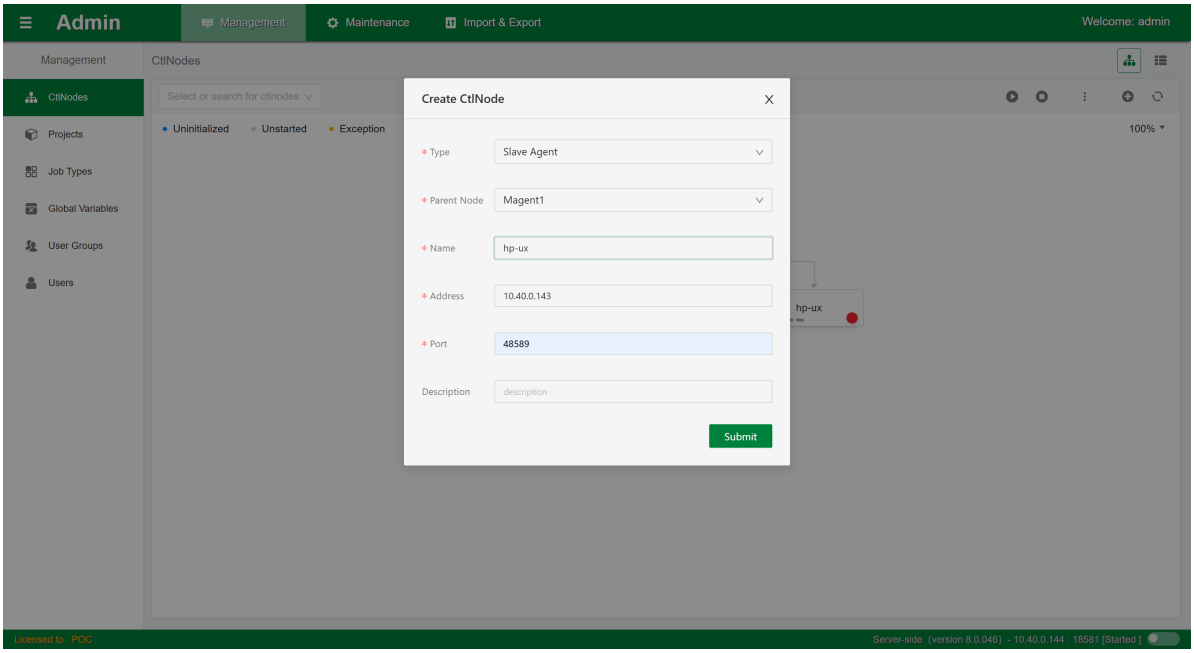
  1. Click the `+` button
  2. Select `Master Agent` for node type
  3. Fill in the node name, IP, port, and select the defined scheduling server node (svrnode) as the superior node
  4. Save the configuration



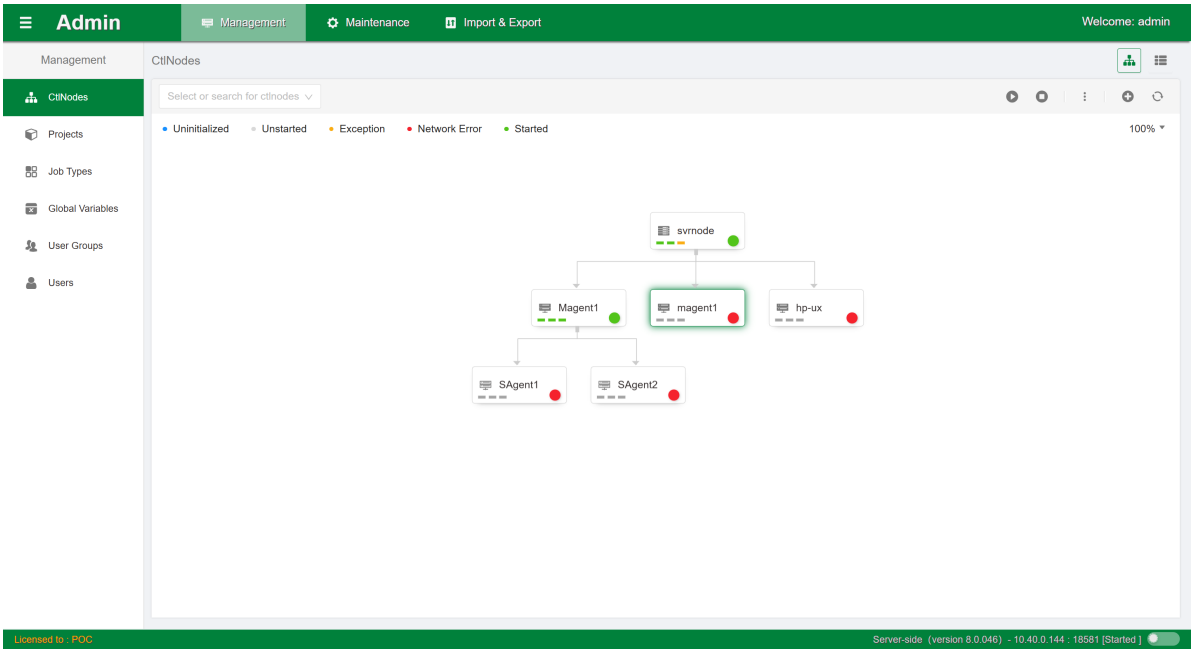
- **Define slave agent node**

  Slave agent nodes run attached to master agent nodes, so you need to define the master agent node first:

  1. Click the `+` button and select `Slave Agent` for type
  2. Fill in the name, IP, port, and select the corresponding master agent node as the superior node
  3. Save the configuration

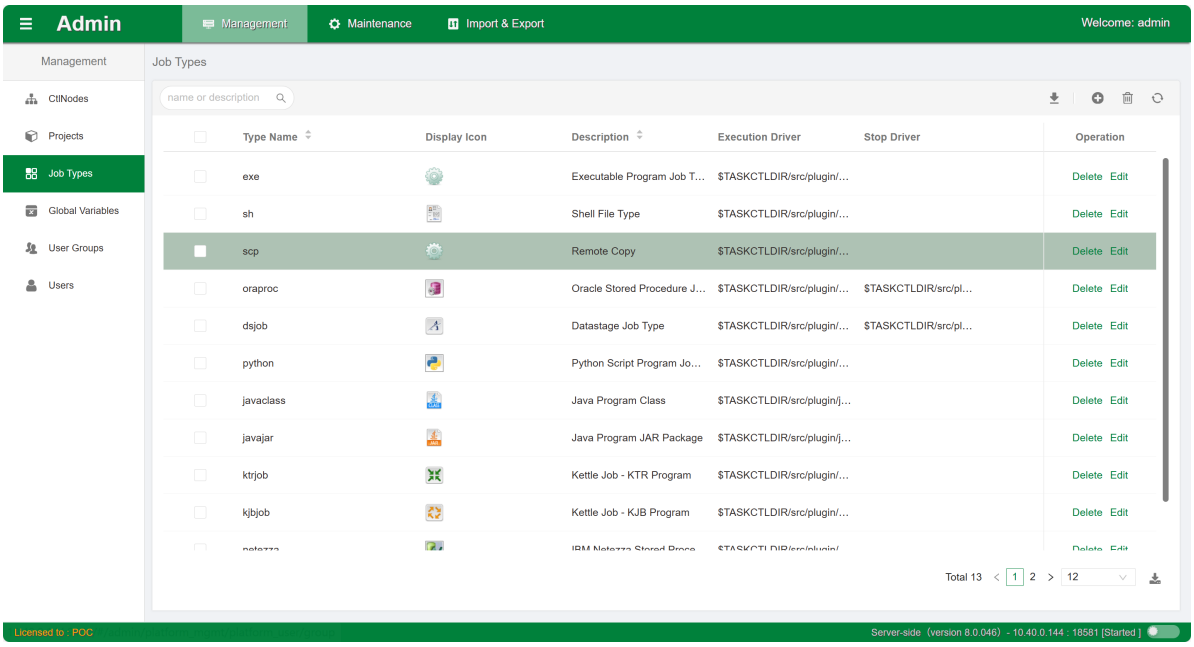


After node definition is completed, you can intuitively view the hierarchical structure and association relationship in the `Node Relationship Diagram`.



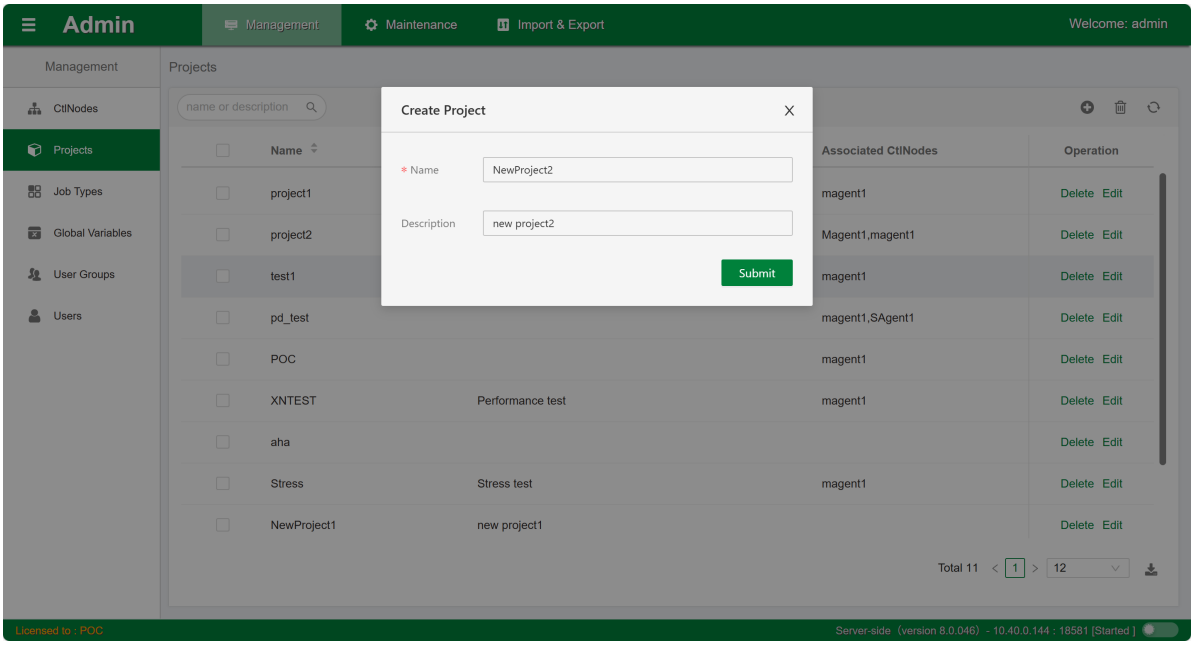
Job Type Definition

Complete job type configuration on the `Job Type` page of the Admin module in the B/S client. You can finely define plugins, parameters, icons, return values, etc. Click `+` to add a new job type.



Application Project Definition

Process development relies on the project environment, so application project definition needs to be completed before development, which can be configured in the Admin module of the B/S client.



Click the `+` button on the corresponding page, configure the project name and project description to complete the definition.

The platform initialization configuration information is uniformly saved to the server-side `conf/ctlconf.xml` file.

Platform Kernel Configuration

This section is not a necessary option for initialization and configuration. Generally, no modification is required; if modification is needed, please operate with caution.

Path of the platform kernel configuration file: `conf/kernel.cfg`

Configurable items: Core parameters such as scheduling capacity, throughput, system message interface, process signal management, and number of days to save historical records.

Note: Some configuration items require restarting the platform after modification, while others do not.

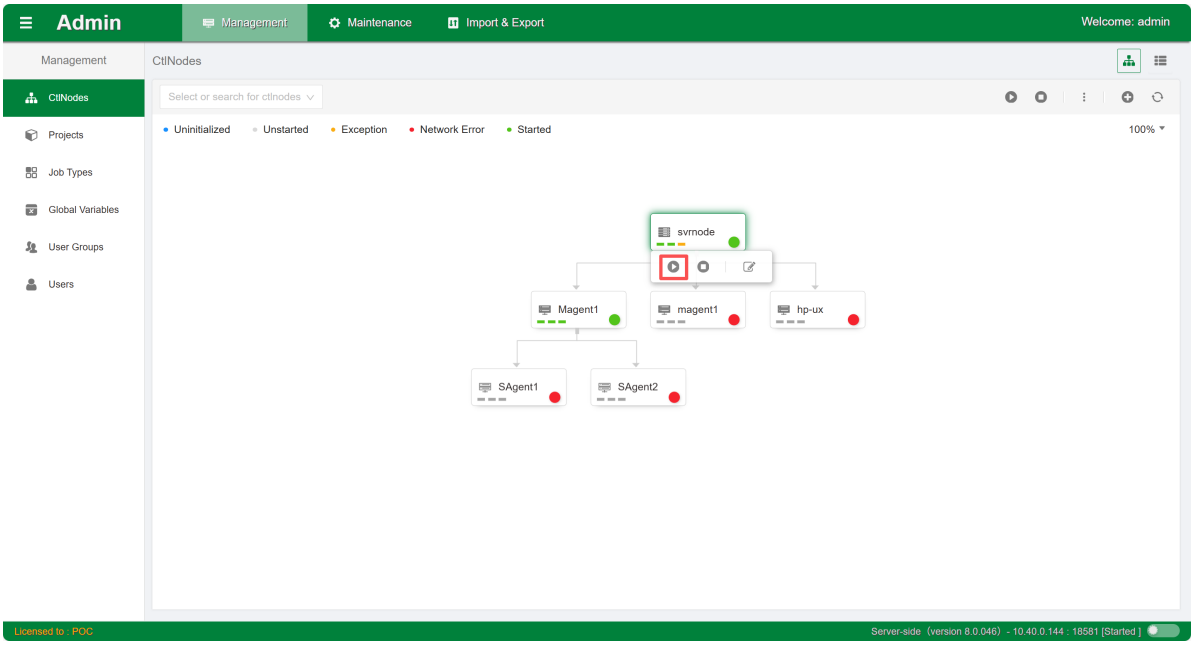
Installation Verification and Health Check

Start Service Node

The core of installation verification is whether the scheduling server can start normally, and three startup methods are provided:

B/S Application Admin Startup

Enter the node relationship diagram of the `Platform Nodes` in the Admin module, select the target node, and click the `Start` button; click the `Stop` button to stop (the node must have completed initialization).



Background Application ctladmin Startup

Log in to the character client `ctladmin` and use `ctlstart <node number>` to start the node. You can query the node number through `listcirn`:

```
~~~~~~~~~~~~~~~~~~~~~~  Official website: www.wloadctl.com
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
** System prompt **
    Use the 'help' command to get more information about all commands.
ADMIN> listcirn

  No  Node-name     Version  Node-type      Install-user  System-type  Node-
status         Handshake-status
-------------------------------------------------------------------------------
----------
   2  svrnode       8.0.046  2-Server node  cir80_en      linux        11-
Started          2-Handshake Successful
   6  Magent1       8.0.046  3-Master Agent cir80_enuse   linux        11-
Started          2-Handshake Successful
   9  magent1                3-Master Agent                            4-Comm
error        0-Handshake Not Initiated
  10  SAgent1                4-Slave Agent n                           4-Comm
error        0-Handshake Not Initiated
  11  SAgent2                4-Slave Agent n                           4-Comm
error        0-Handshake Not Initiated
  12  hp-ux                  3-Master Agent                            4-Comm
error        0-Handshake Not Initiated
-------------------------------------------------------------------------------
---------
                                                    total: 6
ADMIN> ctlstart 10

Start 4-Slave Agent node'SAgent1' ... ...
Return message: Failed to connect sock server [10.0.0.1:11111].
```

Use the `ctlstop` command to stop the node, which has the same usage as `ctlstart`.

Terminal Command Line Startup

Directly execute `ctlstart` in the terminal to start the current node, and `ctlstop` to stop the current node:

```
[wloadctl@node-144-EN ~]$ ctlstart
Starting the Server node ...
 [0001/0001] Loading TCC project1_Flow1.
 1 TCC(s) loaded. Starting other components. Please wait a few seconds...
Server node startup completed.

[wloadctl@node-144-EN ~]$ ctlstop
Begin stopping svrnode (Server node)...
Stoped Successfully!
```

Service and Process Verification

After installation is completed, you can verify the effectiveness of the deployment through system commands.

Server and Agent Verification

Execute `ctlninfo 1`, and the status displayed as `2-Stopped` indicates that the installation is normal:

```
ctlninfo 1

linux@wloadctl CtlNode 'svrnode' basic information:
-------------------------------------------------------------------------------
version        : linux(64) 8.0.046     | id             : 2
type           : 2-Server node         | desc           : Server Node
state          : 2-Stopped             | statedesc      :
ip(domain)     : 10.40.0.144           | port           : 48581
isactive       : Activated             | hasHA          : No HA
svrbegintime   :                       | hisdispatchnum : 0
acttime        :                       | currdispatchnu : 0
searchinteval  : 0                     | fdcsleep       : 0
-------------------------------------------------------------------------------
runmaxnum      : 100                   | currrunnum     : 0
currffnum      : 0                     | maxvirresval   : 1000
currvirresval  : 0                     | currffvirres   : 0
cpu_user       : 3.25                  | memtotal       : 15.36
memused        : 68.99                 | memthreshold   : 100.00
diskuse        : 47.00                 | diskthreshold  : 97.00
-------------------------------------------------------------------------------
```

Application Server Verification

Execute the status check in the Web application directory, and `running` displayed indicates that the service is normal:

```
[wloadctl@node-144-EN wloadctl-service-8.2]$ ./service.sh status
workloadctl is running...
[wloadctl@node-144-EN wloadctl-service-8.2]$
```

Function Verification

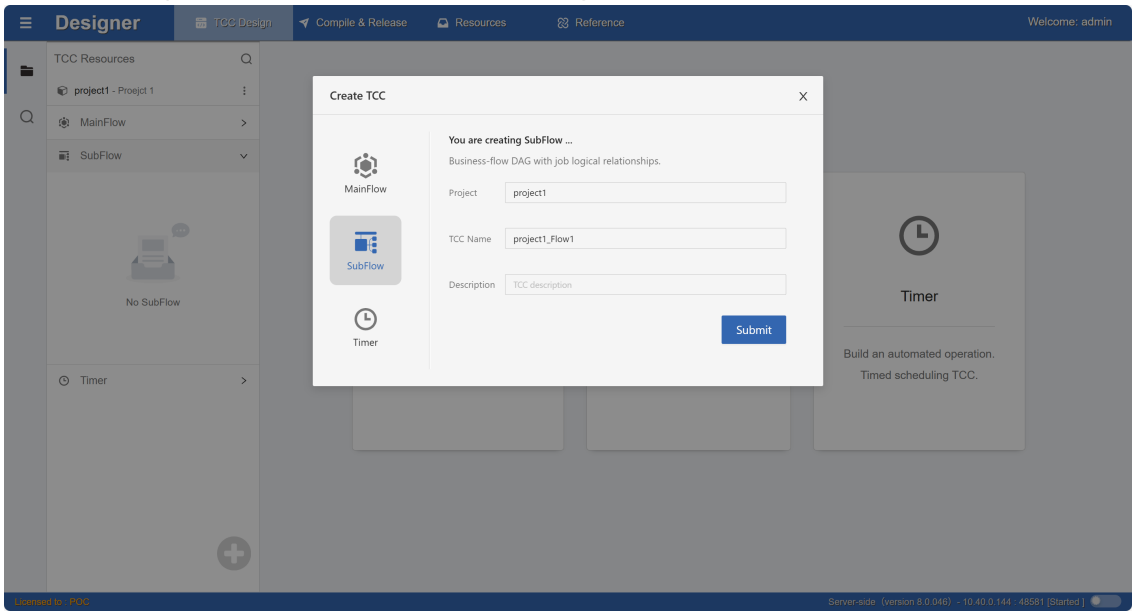
Verify the availability of system functions through the minimum executable scheduling task.

Verification Objectives

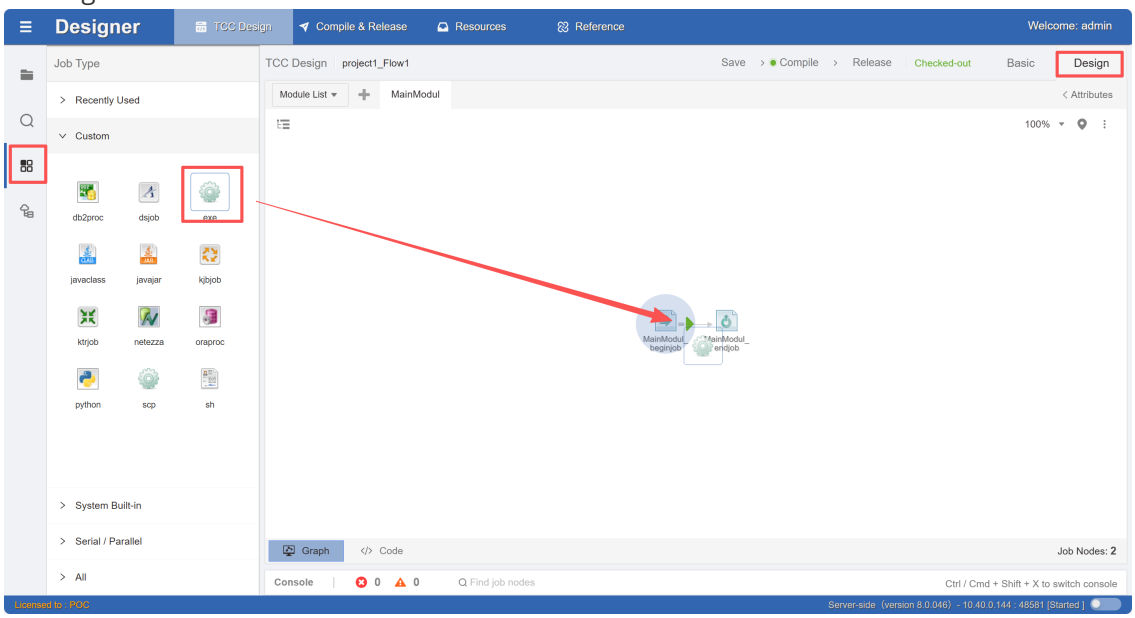
- Verify that the scheduling service can issue tasks normally
- Verify that the Agent can receive and execute tasks normally
- Verify that the execution results can be returned and displayed normally

Verification Steps

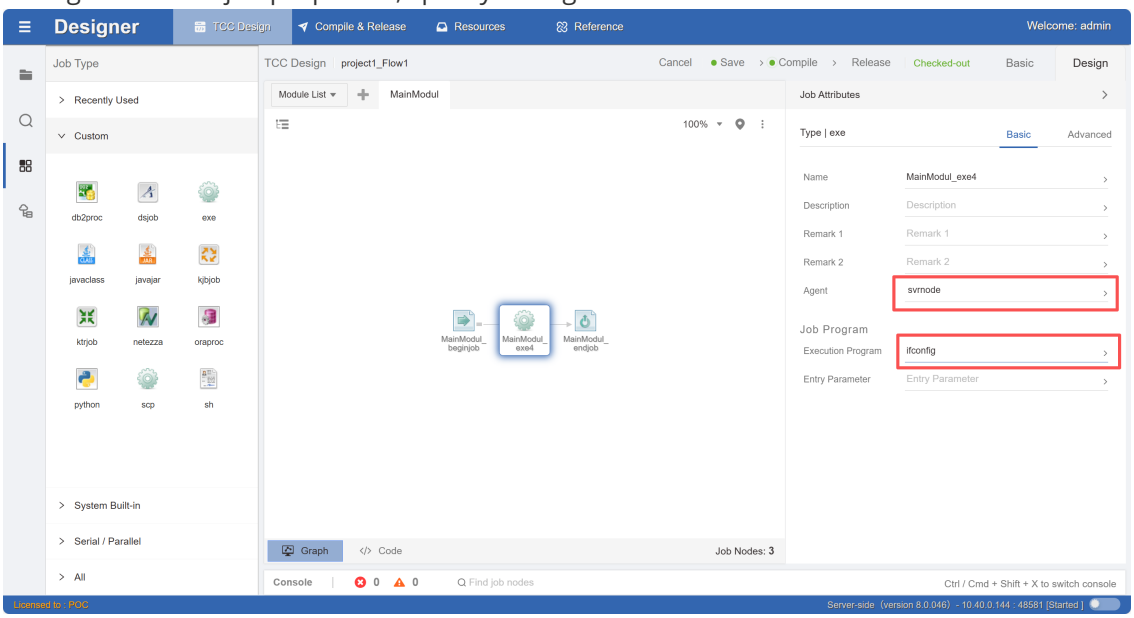
1. Create a test process `projectr1_Flow1` in Designer



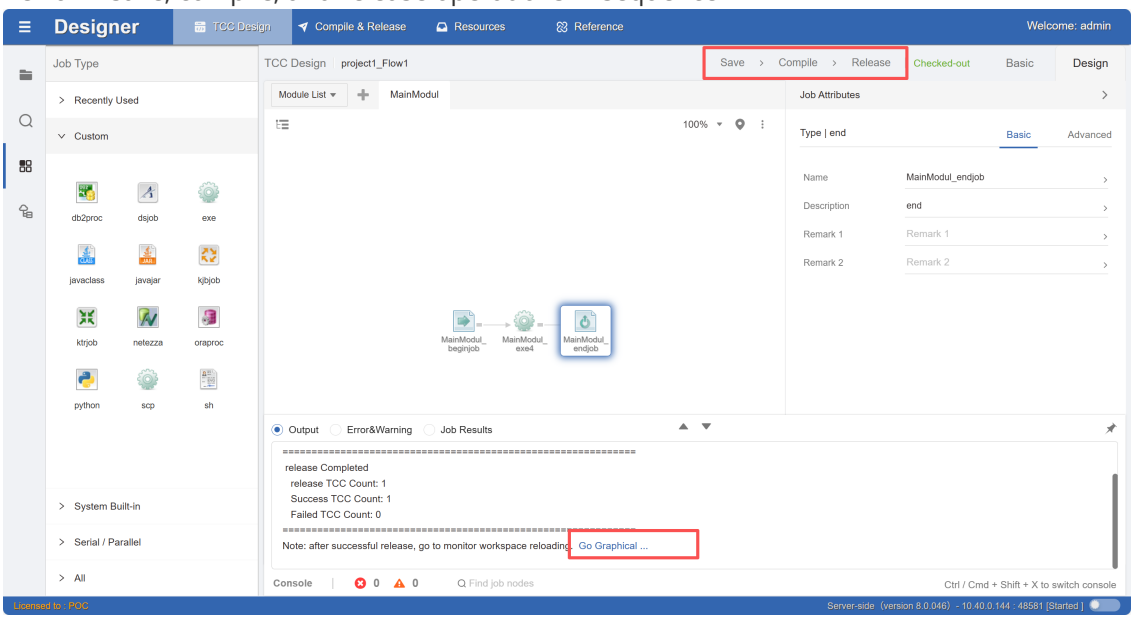
2. Switch to the process design interface, drag the `exe` job from the component library to after the begin node



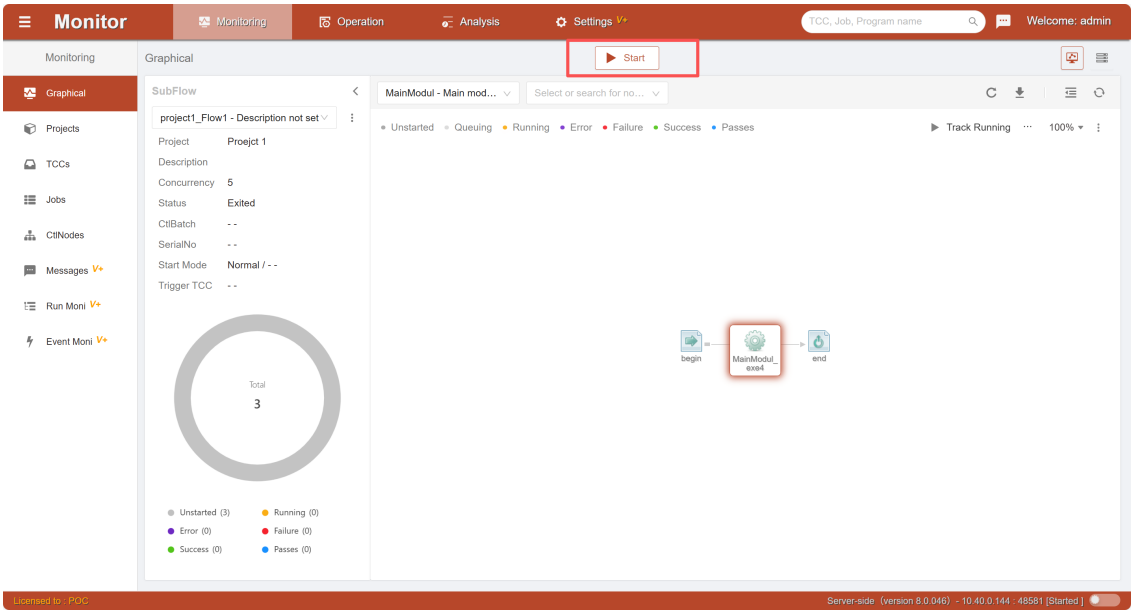
3. Configure the exe job properties, specify the Agent and execution command



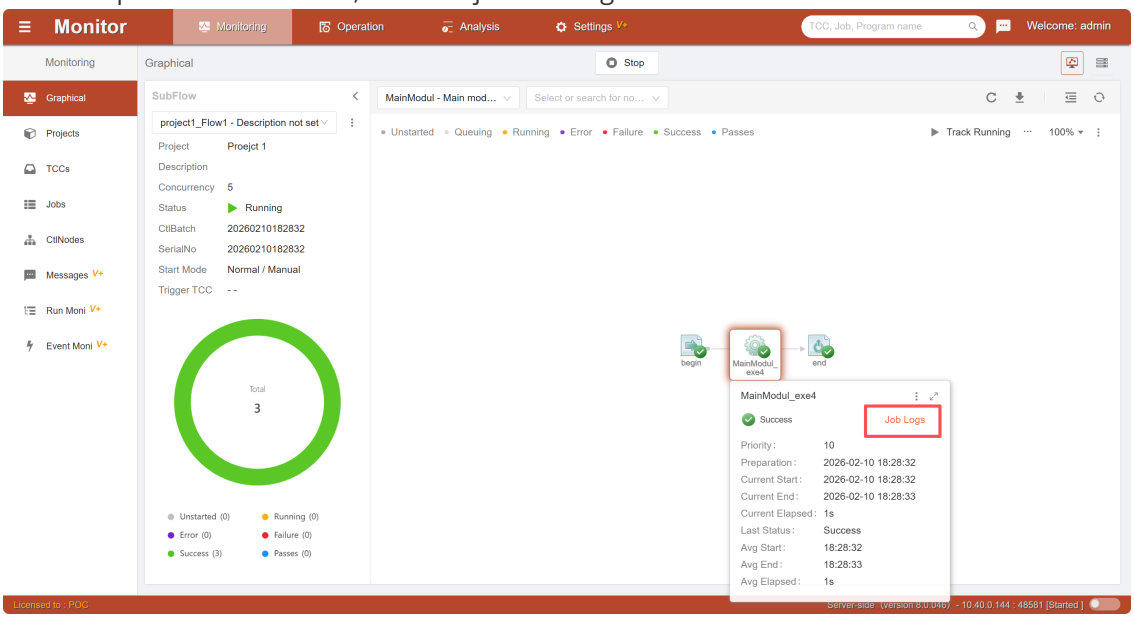
4. Perform save, compile, and release operations in sequence



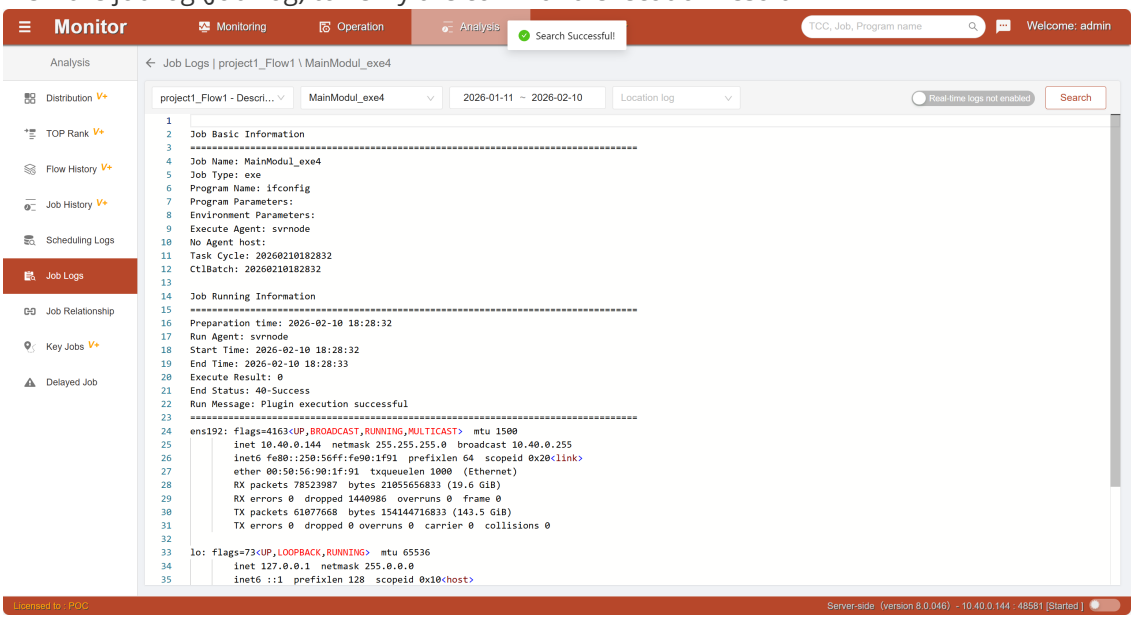
5. Jump to Monitor and click `Start` to execute the process



6. After the process is executed, view the job running status



7. View the job log (Job Log) to verify the command execution result



Platform Deployment Examples

Standalone Deployment

Standalone deployment means the platform deploys only a single scheduling service node without additional agent nodes; the character interface application can be deployed on the same server as the service node, making it the simplest deployment mode.

Environment Preparation

Server: CentOS 8, IP: 10.40.0.144, Open ports: 18581, 8088

Create the system user wloadctl, configure the environment variable TASKCTLDIR and make it effective, and prepare the JDK 17 environment:

```
[root@node-144-EN ~]# useradd wloadctl
[root@node-144-EN ~]# su - wloadctl
[wloadctl@node-144-EN ~]# echo export TASKCTLDIR=\$HOME/taskctl >>
~/.bash_profile
[wloadctl@node-144-EN ~]# echo export PATH=\$PATH:\$TASKCTLDIR/bin:. >>
~/.bash_profile
[wloadctl@node-144-EN ~]# source ~/.bash_profile
[wloadctl@node-144-EN ~]# java -version
java version "17.0.18"
Java(TM) SE Runtime Environment (build 17.0.18+8-LTS)
```

Deployment Method

Refer to the installation steps of the scheduling service node, select **1 Classic Installation**, and the system will automatically package and install the server and background applications. The default IP is the local IP, and the default port is 18581; just press Enter to confirm.

```
--------------------------------------------------------------------------
-              Character interface client installation.                  -
--------------------------------------------------------------------------
Start unpack ... ...
Unpack complete
Start copying files to product directory.... ...
Copy files to product directory completed.
Start cleaning up temporary installation files on client...
Client temporary installation file cleanup completed.
--------------------------------------------------------------------------
-                          Server node Installation.                     -
--------------------------------------------------------------------------
Start unpack ...
Unpack complete.
Start copying files to product directory...
Copy files to product directory completed.
Start cleaning up temporary installation files on Server node...
Server node temporary installation file cleanup completed.

Enter installation configuration information:
--------------------------------------------------------------------------
Input the IP address of the current Server node. [Default is 10.40.0.144]:
Input the port of the current Server node. [Default is 18581]:

Your current input as follows:
--------------------------------------------------------------------------
  Current Server node IP                  :10.40.0.144
  Current Server node PORT                :18581
--------------------------------------------------------------------------
```

Classic installation will prompt whether to install the sample project; it is recommended to select yes for initial deployment.

For Web application deployment, refer to the chapter "Client and Web Service Installation":

```
sh install.sh
**************************************************************************
********************   WLOADCTL V8.0 Online Installer   ***************
**********************       Powered by wloadctl.com       **************
```

```
*************************************************************************
Existing WLOADCTL server installation detected. Using the current server IP and
port.
Enter WLOADCTL Server IP (default: 10.40.0.144): 10.40.0.144
Enter WLOADCTL Server Port (default: 18581): 18581
Enter WLOADCTL Web Application IP (default: 10.40.0.144): 10.40.0.144
Enter WLOADCTL Web Application Port (default: 8088): 8088
*************************************************************************

WLOADCTL Server IP        : 10.40.0.144
WLOADCTL Server Port      : 18581
WLOADCTL Web App IP       : 10.40.0.144
WLOADCTL Web App Port     : 8088
*************************************************************************
Confirm installation? [Y/n]: y
Installation completed successfully!
Please run ./service.sh start
Access URL: http://10.40.0.144:8088/
```

Distributed Deployment

Distributed deployment requires additional deployment of agent nodes or agent clusters; agent nodes can be deployed on the same server as the service node or independently, suitable for complex network application scenarios.

Environment Preparation

On the basis of the standalone environment (10.40.0.144), add a CentOS 8 server: 10.40.0.142, open the agent port 18589.

Create the system user wloadctl, configure the environment variable TASKCTLDIR and make it effective:

```
[root@node-144-EN ~]# useradd wloadctl
[root@node-144-EN ~]# su - wloadctl
[wloadctl@node-144-EN ~]# echo export TASKCTLDIR=\$HOME/taskctl >>
~/.bash_profile
[wloadctl@node-144-EN ~]# echo export PATH=\$PATH:\$TASKCTLDIR/bin:. >>
~/.bash_profile
[wloadctl@node-144-EN ~]# source ~/.bash_profile
```

Deployment Method

Install the agent node: select **2 Custom Installation** → **1 Agent node Installation**, the default IP is the local IP, and the default port is 18589; just confirm directly.

```
-------------------------------------------------------------------------
-                         Agent node installation.                      -
-------------------------------------------------------------------------
Start unpack ... ...
Unpack complete
Start copying files to product directory.... ...
Copy files to product directory completed.
Start cleaning up temporary installation files on Agent node...
Agent node temporary installation file cleanup completed.
```

```
Input installation configuration information:
--------------------------------------------------------------------------
Input currently installed agent node IP[Default is 10.40.0.142]:
Input currently installed agent node PORT[Default is 18589]:


Your current input as follows:
--------------------------------------------------------------------------

  Agent node IP      :10.40.0.142
  Agent node PORT    :18589

--------------------------------------------------------------------------
```

After completing the installation of the agent node instance, you also need to add the agent node on the platform node management page of the Admin function module in the B/S application.

After successful addition, you need to click `Play` to start the node. A green icon for the node status indicates successful deployment.

In addition, if you need to deploy an agent cluster, just install other agent node instances according to this method, then create slave agent nodes and mount them under `magent_142` to complete the deployment of sub-agents and clusters.

After the deployment of distributed nodes (clusters) is completed, you can set the `agent` attribute of the job to `magent_142` (agent name) or `[magent_142]` (cluster name) to specify that the job runs on the agent/cluster.

High Availability Deployment

High availability deployment requires deploying a primary service node and a standby service node, which need to be deployed on different physical hosts in the production environment.

Environment Preparation

Prepare two hosts:

- 10.40.0.144 (Primary service node)
- 10.40.0.142 (Standby service node)

Refer to the chapter "Core Scheduling Service Node Installation" to configure users and environment variables, which will not be repeated here.

Deployment Method

1. Install Primary Service Instance

Refer to the core scheduling service node installation process, select the installation option `2. Custom Installation`, and select `3. Server node (HA Primary)` in the subsequent interface.

- Local service node IP (default): 10.40.0.144, Port (default): 18581
- Standby service node IP (required input): 10.40.0.142, Port (default): 18581

```
--------------------------------------------------------------------------
-                          Server node Installation.                     -
--------------------------------------------------------------------------
```

```
Start unpack ...
Unpack complete.

Start copying files to product directory...
Copy files to product directory completed.

Start cleaning up temporary installation files on Server node...
Server node temporary installation file cleanup completed.

Enter installation configuration information:
--------------------------------------------------------------------------------
Input the IP address of the current Server node. [Default is 10.40.0.144]:
Input the port of the current Server node. [Default is 18581]:
Enter the IP address of the high-availability Standby Server: 10.40.0.142
Enter the port of the high-availability Standby Server [Default is 18581]:

Your current input as follows:
--------------------------------------------------------------------------------
  Current Server node IP                  :10.40.0.144
  Current Server node PORT                :18581
  Standby Server IP                       :10.40.0.142
  Standby Server PORT                     :18581
--------------------------------------------------------------------------------
```

2. Install Standby Service Instance

Refer to the core scheduling service node installation process, select the installation option `2. Custom Installation`, and select `4. Server node (HA Standby)` in the subsequent interface.

- Local service node IP (default): 10.40.0.142, Port (default): 18581
- Primary service node IP (required input): 10.40.0.144, Port (default): 18581
- It is recommended to enter `y` to enable the automatic takeover function

```
--------------------------------------------------------------------------------
-                          Server node Installation.                           -
--------------------------------------------------------------------------------

Start unpack ...
Unpack complete.

Start copying files to product directory...
Copy files to product directory completed.

Start cleaning up temporary installation files on Server node...
Server node temporary installation file cleanup completed.

Enter installation configuration information:
--------------------------------------------------------------------------------
Input the IP address of the current Server node. [Default is 10.40.0.142]:
Input the port of the current Server node. [Default is 18581]:
Enter the IP address of the high-availability Primary Server: 10.40.0.144
Enter the port of the high-availability Primary Server [Default is 18581]:
If the Primary Server crashes unexpectedly, should the Standby Server
automatically take over? y

Your current input as follows:
```

```
--------------------------------------------------------------------------------
  Current Server node IP                  :10.40.0.142
  Current Server node PORT                :18581
  Primary Server IP                       :10.40.0.144
  Primary Server PORT                     :18581
  If the Primary Server crashes unexpectedly, should the Standby Server
automatically take over   :y
--------------------------------------------------------------------------------
```

3. Start Primary and Standby Service Nodes

The startup process of the active-standby mode is different from that of the single-service mode; you need to initialize port listening first, then activate the node:

```
# 1. Initialize port listening (execute on both primary and standby nodes)
[wloadctl@node-142-EN cirinst]$ ctlinit

Server node begins initialization....
The listening port of the Server is: 18581
Initialization successful. Current released version: 8.0.046, compiled version:
20260211171848

# 2. Activate and start the service on the primary service node
[wloadctl@node-144-EN cirinst]$ ctlinit
[wloadctl@node-144-EN cirinst]$ ctlactsvr

Start validating servre node [10.40.0.144:48581]...
Start validating the remote server standby machine [10.40.0.142:48581]...
Current host start activating...
The current host activation successful

Wait for agent node handshake...
-------------------------------[ Notice ]--------------------------------
    The current host is now activated. You can start the current host
 instance immediately, or run the 'ctlstart' command later to start
 the service instance.
--------------------------------------------------------------------------------
Should the current host instance be started immediately? (y/n)y

Starting the Server node ...
 0 TCC(s) loaded. Starting other components. Please wait a few seconds...
Server node startup completed.
```

4. Configure Automatic Switchover for B/S Application Service

Modify `config/application-prod.yml` (Note: Fix the spelling error in the original document from `applicaiton-prod.yml` to `application-prod.yml`):

```yaml
taskctl:
  # Scheduling Engine - Primary Node Configuration
  server:
    ip: 10.40.0.144
    port: 18581
  # Scheduling Engine - Standby Node Configuration
  server-slave:
    # Enable standby node (default false)
    enable: true
    ip: 10.40.0.142
    port: 18581
```

Restart the application to make the configuration take effect:

```
[wloadctl@node-144-EN wloadctl-service-8.2]$ ./service.sh restart
Stopping workloadctl...
workloadctl (pid:4025754) is exiting...
workloadctl has exited.
Started workloadctl successfully...
Java HotSpot(TM) 64-Bit Server VM warning: ignoring option PermSize=256M;
support was removed in 8.0
...
2026-02-11T18:40:13.810+0800: [GC (Allocation Failure) [PSYoungGen: 253920K-
>8160K(253952K)] 291056K->60125K(516096K), 0.0143580 secs] [Times: user=0.09
sys=0.00, real=0.01 secs]
18:40:14.467 [main] INFO  org.springframework.boot.SpringApplication - Started
application in 14.023 seconds (JVM running for 16.979)
```

Product Registration

Registration Statement

Authorization Status	Description of Function Restrictions
Over Authorization	Core functions remain unchanged, application experience functions are reduced, can be normally used in production systems
Unregistered	Does not support 7×24-hour continuous automated scheduling, prohibited for use in production environments
Basic Authorization Edition	Permanent use, supports 7×24-hour automated operation, includes basic scheduling core + display functions, some advanced functions are restricted
Advanced Authorization Edition	All functions of the basic edition + data analysis (Monitor, etc.), free trial for 3 months, reverts to basic edition upon expiration
Commercial Edition	All functions of the advanced edition + optional additional components (number of jobs/nodes/parallelism, etc.), supports enterprise secondary development

Registration Steps

1. Obtain Platform Installation ID

When unregistered, logging in to the B/S client or character interface will display the `installation ID`:

```
ctladmin

CTL*Admin: WLOADCTL Platform comprehensive management software.
Verion       : 8.0.046
Release      : Feb  3 2026 15:49:13
Copyright    :  Task Information Technology Co Ltd.
Licensed to  : WLOADCTL
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~
Server IP [default: '10.40.0.144']              :
Server PORT [default: '48581']                  :
Login Username                                  : admin
Login Password                                  :

               ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                              Software not registered
                      Current installation ID is: 009K.6RL1.03RC.U2L4.00YY
                      Please visit www.wloadctl.com to obtain authorization
               ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~  Official website: www.wloadctl.com  ~~~~~~~~~
[Unregistered] ~~~~~~~~~
** System prompt **
    Use the 'help' command to get more information about all commands.
ADMIN>
```

2. Obtain Authorization License

- Offline: Obtain your license by contacting our support team via service@wloadctl.com or Microsoft Teams (kitleer@gmail.com). Provide your Installation ID and select the product specification (advanced edition for free trial, commercial edition requires payment).

3. Complete Registration

Method 1: Character Interface

```
ADMIN> regist
Please input licence file name or LICKEY:POC_20261231.lic
Registration successful!
ADMIN> quit

# Verify registration result
[wloadctl@node-144-EN ~]$ ctladmin
# After login, the [Registered] identifier is visible, and there is no
unregistered prompt
```

Method 2: B/S Client

Enter the license in the pop-up registration window and click `Submit`.

Upgrade, Rollback and Uninstallation

This chapter only applies to server/agent operations, excluding application server operations.

Version Number Description

Core service version number = Released version number + Compiled version number, which can be viewed by executing `ctlinit`:

```
[wloadctl@node-144-EN cirinst]$ ctlinit
Server node begins initialization....
The listening port of the Server is: 48581
Initialization successful. Current released version: 8.0.046, compiled version:
20260211144617
```

- Released version number (e.g., 8.0.046): Changes may involve core data structures;
- Compiled version number (e.g., 20260211144617): Only code compilation updates, no data structure changes.

Backup Before Upgrade

Scheduling core data is stored as files; just back up the entire product directory (excluding large directories such as history):

```
cd $TASKCTLDIR/ctlvm
# Exclude the history directory to avoid excessive backup size
tar -zcvf wloadctl_8.0.046_20260212.tar.gz --exclude='wloadctl_8.0.046/history'
wloadctl_8.0.046
```

Upgrade Steps

1. One-click Platform Upgrade (Only Used When Released Version Number Changes)

```
[wloadctl@node-144-EN ctlvm]$ ctlupgrade 1

============= Upgrade Started: 2026-02-12 10:43:47 ================
Start to get basic information of all background nodes... ...
Start to verify network connectivity of 1 background nodes... ...
Enter the current upgrade package name ['linux' version]:
cir_linux_8.0.046_amd64_20260212.tar.gz
Enter the current upgrade package name ['aix' version]:
cir_aix_8.0.046_amd64_20260212.tar.gz
```

⚠ Note: Ensure network connectivity of all nodes, as data structure conversion takes a long time.

2. Selective Node Upgrade (Commonly Used)

Step 1: Configure Node-Upgrade Package Mapping

```
cd $TASKCTLDIR/upgrade
vi deploy.conf
# Format: Node name  Upgrade package name
magent_142      cir_linux_8.0.046_amd64_20260212.tar.gz
```

Step 2: Upload Upgrade Package to Specified Directory

```
# Upload directory: upgrade/deploypkg
[wloadctl@node-144-EN deploypkg]$ pwd
/home/wloadctl/taskctl/upgrade/deploypkg
[wloadctl@node-144-EN deploypkg]$ ll
total 30628
-rw-r--r-- 1 wloadctl wloadctl 31360613 Feb 12 11:03
cir_linux_8.0.046_amd64_20260212.tar.gz
```

Step 3: Execute Upgrade Command

```
[wloadctl@node-144-EN ~]$ ctlupgrade 0

============= Upgrade Started: 2026-02-12 11:13:40 ================
Start to get basic information of all background nodes... ...
Start to verify network connectivity of 1 background nodes... ...
Starting to upload installation packages to 1 nodes... ...
No:0000 Installation package is being uploaded to node
'magent_142'[10.40.0.142:18589]
Node installation package transferred successfully
'magent_142'[10.40.0.142:18589]
Installation package upload completed for 1 nodes
Starting to check installation packages for 1 nodes... ...
Installation package check completed for 1 nodes

Starting core agent node stop processing... ...
No:0000  Agent 'magent_142' starting to stop...
No:0000 Agent 'magent_142' stopped successfully!
1 agent nodes stopped successfully

Starting node upgrade
No:0000 Starting upgrade of node magent_142[10.40.0.142:18589], from version
'8.0.046' to target version '8.0.046'
Node magent_142[10.40.0.142:18589] upgrade successful, elapsed time 11 second(s)

Starting reinitialization of all nodes... ...
Starting all execution agent instances... ...
No:0000 Node magent_142[10.40.0.142:18589] reinitialization command has been
sent

Waiting for each node to initialize, this may take 5~10 minutes, please wait
patiently... ...
  Nodes such as 'magent_142' are waiting for initialization...
  Nodes such as 'magent_142' are waiting for initialization...
Node 'magent_142'[10.40.0.142:18589] reinitialization succeeded!
```

```
All nodes reinitialized successfully!

No:0000  Starting agent node 'magent_142'[10.40.0.142:18589]
No:0000 Agent 'magent_142' started successfully!
1 agent nodes started successfully
============= Upgrade Completed: 2026-02-12 11:15:32 ================
```

Rollback Strategy

1. Rollback Prerequisites

- Full backup before upgrade has been completed (core: `$TASKCTLDIR/ctlvm` directory);
- All nodes to be rolled back are in **stopped state** (execute `ctlstop + ctlshut`).

2. Rollback Steps

Step 1: Stop Nodes to Be Rolled Back

```
# Universal stop command for primary/agent nodes
[wloadctl@node-144-EN cirinst]$ ctlstop
Begin stopping svrnode (Server node)...
Stoped Successfully!

[wloadctl@node-144-EN cirinst]$ ctlshut
CtlNode shutdown successfully.
```

Step 2: Restore Backup Files

```
# Enter the ctlvm directory, delete the upgraded directory, and restore the
backup
cd $TASKCTLDIR/ctlvm
rm -rf wloadctl_8.0.046
tar -zxvf wloadctl_8.0.046_20260212.tar.gz
```

Step 3: Re-initialize and Start Nodes

```
# Initialize port listening
[wloadctl@node-144-EN cirinst]$ ctlinit
# Start the node (if using active-standby mechanism, the primary node needs to
execute ctlactsvr to activate)
[wloadctl@node-144-EN cirinst]$ ctlstart
```

3. Verify Rollback Result

Execute `ctlinit` to view the version number, confirm that it is restored to the released version number + compiled version number before the upgrade, and ensure that the business runs normally after the node is started.

Service Node Uninstallation

1. Uninstallation Prerequisites

Ensure the node is completely stopped:

```
# Stop the service
[wloadctl@node-144-EN cirinst]$ ctlstop
Begin stopping svrnode (Server node)...
Stoped Successfully!

# Shut down the node
[wloadctl@node-144-EN cirinst]$ ctlshut
CtlNode shutdown successfully.

# (Optional) Additional confirmation of process stop for Linux/AIX systems
# Linux
[wloadctl@node-144-EN cirinst]$ sh killlix.sh
# AIX
[wloadctl@node-144-EN cirinst]$ sh killaix.sh
```

2. Execute Uninstallation

```
# Delete the entire product directory (irreversible, confirm backup is
completed)
[wloadctl@node-144-EN cirinst]$ rm -rf $TASKCTLDIR
```

Security and Production Deployment Recommendations

Principle of Least Privilege

- **System Accounts**: Create dedicated runtime accounts (e.g., `wloadctl`), prohibit running services with `root/administrator`;
- **File Permissions**: Set the installation directory to read-only, and only the runtime account has read/write permissions for data/log directories;
- **Application Permissions**: Design permissions for platform projects according to "minimum access", and strictly control the scope of user operations.

Network Isolation Recommendations

- **Port Minimization**: Only open ports necessary for business, close useless listening;
- **Source Restriction**: Only allow intranet/trusted IPs to access management/console ports;
- **Domain Isolation**: Divide application services, databases, and scheduling nodes into different security domains, and restrict cross-domain access.

Audit and Logs

- **Log Content**: Record key behaviors such as login, task start/stop, configuration modification, exceptions, and unauthorized access;
- **Log Management**: Roll and split by size/time, retain for ≥90 days to prevent disk full;
- **Audit Capability**: Support multi-dimensional retrieval, strictly control log file permissions to prevent tampering/deletion.