**For support project**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

For a **Unix interview in a support project**—especially involving tools like **Control-M**, **file processing**, and **shell scripting**—you can expect a mix of scenario-based and technical questions. Here's a breakdown of key areas and sample questions to help you prepare:

**🔧 1. Control-M (Job Scheduling Tool)**

Control-M is widely used for batch job scheduling and monitoring. Interviewers may ask:

**Conceptual Questions**

* What is Control-M and how is it used in your project?

: **Control-M** is a workload automation and job scheduling tool developed by BMC Software. It's widely used in enterprise IT environments to manage and automate batch jobs across various platforms and applications.

**🔧 What is Control-M Used For?**

Control-M helps organizations:

**Automate workflows** across different systems (mainframes, databases, cloud platforms, etc.)

**Schedule jobs** to run at specific times or based on events

**Monitor and manage** job execution in real-time

**Handle dependencies** between jobs and systems

**Improve reliability** and reduce manual intervention

* **🧩 How Is Control-M Used in Projects?**
* In a typical project, Control-M might be used for:
* **Batch Job Scheduling**  
  Automating daily, weekly, or monthly jobs like data extraction, transformation, and loading (ETL), report generation, or backups.
* **Workflow Orchestration**  
  Linking jobs across systems (e.g., SAP, Oracle, Hadoop, AWS) to ensure they run in the correct sequence.
* **Error Handling & Notifications**  
  Setting up alerts for job failures or delays, and triggering recovery actions or notifications.
* **Audit & Compliance**  
  Maintaining logs and reports for job execution history, which helps in audits and troubleshooting.
* **🏗️ Example in a Project Context**
* Let’s say your project involves processing customer data every night:
* **Step 1:** Control-M triggers a job at 2 AM to extract data from a CRM system.
* **Step 2:** Once extraction is complete, it runs a transformation job to clean and format the data.
* **Step 3:** After transformation, it loads the data into a reporting database.
* **Step 4:** If any step fails, Control-M sends an alert to the support team and may retry or run a fallback job.
* **What are the different job types in Control-M?**

Control-M supports a variety of job types to automate and manage different kinds of tasks across platforms and applications. Here's a breakdown of the most commonly used job types:

🔧 Common Job Types in Control-M

| Job Type | Description |
| --- | --- |
| Command Job | Executes shell scripts or command-line instructions on target servers (Linux/Unix/Windows). |
| File Transfer Job | Automates file transfers between servers using protocols like FTP, SFTP, FTPS. |
| Database Job | Runs SQL queries or stored procedures on databases like Oracle, SQL Server, MySQL, etc. |
| Application Integrator Job | Custom jobs created using Control-M Application Integrator for non-standard applications. |
| Web Services Job | Invokes REST or SOAP APIs for integration with web services. |
| Informatica Job | Executes workflows in Informatica PowerCenter or Cloud. |
| SAP Job | Runs SAP batch jobs using Control-M for SAP. |
| AWS/Azure/GCP Jobs | Manages cloud resources and workflows (e.g., EC2 start/stop, Lambda execution). |
| Hadoop Job | Executes jobs in big data environments like Hive, Pig, Spark, etc. |
| BIM (Batch Impact Manager) Job | Monitors SLAs and predicts delays in batch processing. |
| File Watcher Job | Monitors file creation/modification and triggers workflows based on file events. |
| Job Flow Job | Defines a sequence of jobs with dependencies and conditions. |
| Embedded Script Job | Contains inline scripts (e.g., Python, Bash) directly within the job definition. |

🧩 Specialized Job Types (via Plugins or Add-ons)

* Control-M for Kubernetes – Manages containerized workloads.
* Control-M for Airflow – Integrates Apache Airflow DAGs into Control-M workflows.
* Control-M for Snowflake – Executes queries and tasks in Snowflake data warehouse.
* Control-M for Salesforce, ServiceNow, etc. – Automates tasks in SaaS platforms.

**-------------------------------------------------------------------------------------------------------**

* **How do you handle job failures in Control-M?**

Handling job failures in Control-M is a critical part of ensuring reliable and automated workload execution. Here's a structured approach to how job failures are typically managed:

🛠️ 1. Automatic Retry Mechanism

Control-M allows you to configure retry settings for jobs:

* You can specify the number of retries and the interval between retries.
* Useful for transient issues like temporary network failures or resource unavailability.

📩 2. Alerts and Notifications

When a job fails:

* Control-M can send alerts via email, SMS, or other integrated notification systems.
* Alerts can include job name, failure reason, and logs for quick diagnosis.

🔁 3. Recovery Jobs

You can define recovery or fallback jobs:

* These are triggered automatically when a job fails.
* For example, if a data load job fails, a recovery job might clean up partial data or notify the support team.

📊 4. Job Logs and Output Analysis

Control-M provides detailed logs and output files:

* These help in identifying the root cause of failure.
* You can view logs directly from the Control-M GUI or CLI.

🔄 5. Conditional Workflows

Using "On-Do Actions", you can define:

* What should happen on success, on failure, or on specific exit codes.
* For example, if Job A fails, skip Job B and trigger Job C to handle the exception.

🔒 6. SLA Management

Control-M can monitor Service Level Agreements (SLAs):

* If a job is delayed or fails, it can escalate the issue based on SLA thresholds.
* Helps in proactive issue resolution before business impact.

👨‍💻 7. Manual Intervention (if needed)

In some cases:

* Jobs may require manual investigation and rerun.
* Control-M allows authorized users to force rerun, hold, or cancel jobs.

✅ Best Practices

* Always set clear error handling rules.
* Use naming conventions and job documentation for easier troubleshooting.
* Regularly review job history and failure trends to improve reliability

**-------------------------------------------------------------------------------**

**Scenario-Based**

* A job is stuck in “Wait Condition” or “Executing” state—how would you troubleshoot?
* How do you set up dependencies between jobs?
* How do you rerun a job that failed yesterday without affecting today's schedule?
* To rerun a job that failed yesterday **without affecting today's schedule**, you’ll want to follow a controlled and isolated approach. Here’s a general strategy that works across most scheduling systems (like Control-M, Autosys, cron jobs, etc.):
* **✅ Steps to Rerun a Failed Job Safely**
* **Identify the Job and Failure Reason**
* Check logs or alerts to understand why the job failed.
* Confirm that the issue has been resolved (e.g., missing file, DB connection, etc.).
* **Check Today’s Schedule**
* Ensure the job hasn’t already run today or isn’t scheduled to run soon.
* If it’s a recurring job, note the next scheduled run time.
* **Use a Manual Rerun Option**
* Most job schedulers allow you to rerun a job manually for a specific date or instance.
* Use the **"rerun"** or **"re-execute"** command with the date/time of the failed run.
* **Isolate the Rerun**
* Run the job in a **test or isolated environment** if possible.
* Use flags or parameters to indicate it’s a rerun (some systems support this).
* **Avoid Triggering Dependencies**
* Disable downstream jobs temporarily if the rerun might trigger them.
* Alternatively, rerun the job with dependency checks turned off.
* **Monitor the Rerun**
* Watch logs and outputs to ensure it completes successfully.
* Validate the output to confirm it matches expectations.
* **Example (Control-M)**
* If you're using Control-M:
* Go to the **Monitoring View**.
* Find the job that failed yesterday.
* Right-click → **Rerun** → Choose the specific date/time.
* Ensure “Do not trigger successors” is selected if needed.

**📁 2. File Processing in Unix**

This includes working with logs, data files, and automation scripts.

**Common Questions**

* How do you check if a file exists and is not empty?
* How do you find and delete files older than 7 days?
* How do you monitor a directory for new files?

**Sample Commands**

Shell

# Check if file exists and is not empty

[ -s filename.txt ] && echo "File exists and is not empty"

# Find and delete files older than 7 days

find /path/to/files -type f -mtime +7 -exec rm {} \;

Show more lines

**🐚 3. Shell Scripting**

Expect questions on writing and debugging shell scripts.

**Topics to Cover**

* Variables, loops, conditionals (if, case)
* Functions and exit codes
* Reading from and writing to files
* Scheduling scripts with cron

**Sample Questions**

* Write a script to check disk space and send an alert if usage exceeds 90%.
* How do you pass arguments to a shell script?
* What is the difference between > and >>?

**🧠 4. Troubleshooting & Support Scenarios**

These test your real-world problem-solving skills.

**Examples**

* A scheduled job didn’t run—how do you investigate?
* A script is failing intermittently—how do you debug?
* How do you handle file permission issues in a shared environment?