Exploration and Practice of Linux Operating System Teaching in the Intelligent Connected Vehicle Major

Hong Zhang, Lin Cheng, Ping Yang
Beijing Polytechnic University, Beijing, China

Abstract: With the rapid development of intelligent connected vehicle (ICV) technology, the automotive industry is undergoing an unprecedented transformation, leading to a surge in demand for highly skilled professionals who possess interdisciplinary knowledge in software, hardware, and network systems. This paper first examines the strategic significance of establishing dedicated academic programs in the ICV major to meet the evolving needs of the industry and cultivate specialized talent. It then delves into the critical role of the Linux operating system within this domain, highlighting its indispensability for developing robust, real-time, and secure vehicular software platforms, including autonomous driving algorithms, infotainment systems, and vehicle-to-everything (V2X) communication modules. The core of this study presents a comprehensive exploration of Linux teaching content tailored for ICV students, covering essential topics such as kernel customization, device driver development, real-time system optimization, and system security. Furthermore, it introduces innovative pedagogical approaches that integrate theoretical instruction with hands-on practical training, project-based learning, and industry collaborations. By sharing these insights and experiences, this paper aims to provide a valuable reference framework for educational institutions seeking to enhance their Linux curriculum and effectively prepare a new generation of engineers for the challenges and opportunities in the intelligent connected vehicle sector.

Keywords: Intelligent Connected Vehicle; Linux Operating System; Teaching Methods; Practical Teaching; Talent Cultivation

1. Significance of Establishing the Intelligent Connected Vehicle Major

1.1 Industry Development Demands

1.1.1 Enormous market potential

As a crucial development direction for the future automotive industry, intelligent connected vehicles integrate cutting-edge technologies such as autonomous driving, Internet of Vehicles, and artificial intelligence, offering vast market prospects.

1.1.2 Driving industrial upgrading

The traditional automotive industry is accelerating its transformation toward intelligence and connectivity, with automakers, technology companies, and component suppliers increasing R&D investments, thereby propelling overall industrial upgrading and transformation. Establishing an intelligent connected vehicle major can supply the industry with specialized talent and facilitate industrial upgrading.

1.1.3 Policy support and guidance

Governments worldwide have introduced policies to support the development of intelligent connected vehicles, including formulating development plans, providing financial support, and constructing test facilities. These policies create a favorable policy environment for the establishment and development of the intelligent connected vehicle major. The development of intelligent connected vehicle operating systems is vital for strengthening China's voice and dominance in the automotive sector, and it carries profound implications for industrial security and national security [1].

1.2 Talent Cultivation Objectives

1.2.1 Demand for interdisciplinary talent

As intelligent connected vehicles are deployed at scale, the industry faces a severe talent shortage, particularly for interdisciplinary technical-skilled professionals and high-caliber innovative talents, resulting in a significant gap [2].

Intelligent connected vehicles involve multiple disciplines including automotive engineering, electronic information, computer science, and communication technology, requiring the cultivation of interdisciplinary talent with cross-domain knowledge and comprehensive practical abilities. Establishing an intelligent connected vehicle major can integrate relevant disciplinary resources, construct a comprehensive curriculum system, and train professionals adapted to industry development needs.

1.2.2 Cultivation of innovation capability

The intelligent connected vehicle field experiences rapid technological updates, requiring talent with strong innovation and problem-solving abilities. By establishing a dedicated major, students can be provided with practical platforms and innovative environments, encouraging participation in research projects and competitions to cultivate innovative thinking and practical skills.

1.2.3 Enhancement of employment competitiveness

With the rapid development of the intelligent connected vehicle industry, employment prospects for relevant professionals are broad. Establishing an intelligent connected vehicle major can enhance students' employment competitiveness, enabling them to smoothly enter automotive manufacturers, technology companies, component suppliers, and related organizations upon graduation to engage in R&D, testing, production, management, and other roles.

2. Importance of the Linux Operating System in the Intelligent Connected Vehicle Major

2.1 Wide Technical Application

2.1.1 In-vehicle infotainment systems

The in-vehicle infotainment (IVI) systems of intelligent connected vehicles are typically developed based on the Linux operating system, such as Android Automotive OS. Linux's open-source nature and flexibility enable it to meet the customization needs of different automakers for IVI systems, providing rich functions and services such as navigation, entertainment, and communication.

2.1.2 Autonomous driving systems

Autonomous driving systems demonstrate the potential to substantially enhance roadway capacity[3]. Autonomous driving technology is one of the core components of intelligent connected vehicles, and its underlying operating systems mostly adopt Linux. Linux's high performance, stability, and real-time capability can meet the requirements of autonomous driving systems for data processing and real-time control, support complex algorithm operation and sensor data fusion, and ensure the safety and reliability of autonomous driving.

2.1.3 Internet of vehicles technology

In the foreseeable future, the Internet of Vehicles (IoV) and Intelligent Transportation Systems (ITS) are poised to undergo synergistic advancement. Concurrently, propelled by the ongoing evolution and refinement of cutting-edge technologies—including 5G communication, the Internet of Things (IoT), and cloud computing—the integration of intelligent connected vehicles with smart city development will be markedly accelerated [4]. IoV is the key technology for achieving interconnectivity among intelligent connected vehicles. The Linux operating system is widely used in IoV devices and servers. It can support multiple communication protocols and network architectures, enabling efficient communication between Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I), providing technical support for building intelligent transportation systems.

2.2 Open-Source Ecosystem Advantages

2.2.1 Resource sharing and collaboration

As an open-source operating system, Linux has a vast developer community and abundant open-source resources. Developers in the intelligent connected vehicle field can leverage the strength of the open-source community to share code, tools, and experience, reducing development costs and improving efficiency. Meanwhile, the open-source ecosystem promotes global technical collaboration and innovation, driving the rapid development of intelligent connected vehicle technology.

2.2.2 Customization and innovation

The open-source feature allows automakers and tech companies to customize and optimize the Linux operating system according to their own needs, developing intelligent connected vehicle software systems that meet specific application scenarios. This customization capability provides space for corporate technical innovation and differentiated competition, helping enhance market competitiveness.

2.2.3 Talent cultivation and employment

Linux's open-source ecosystem provides students majoring in intelligent connected vehicles with abundant learning resources and practical opportunities. Students can participate in open-source projects to deeply understand the principles and applications of the Linux operating system, cultivating practical skills and innovative thinking. Meanwhile, students mastering Linux technology have high competitiveness in the job market, meeting enterprise demands for intelligent connected vehicle technical talent.

2.3 Educational and Practical Value

2.3.1 Foundation for theoretical teaching

The principles and architecture of the Linux operating system constitute an important theoretical foundation for the intelligent connected vehicle major. Linux is stable, highly modular, and open-source, which facilitates the development of customized operating systems by enterprises [5]. By learning Linux students can master basic operating system concepts, process management, memory management, file systems, and other knowledge, laying the foundation for deeply understanding the software system architecture and operational mechanisms of intelligent connected vehicles.

2.3.2 Platform for practical teaching

Linux provides abundant practical teaching resources and experimental environments. Students can practice programming, system configuration, and network setup on Linux systems, cultivating hands-on abilities and problem
-solving skills. For example, by setting up a Linux-based in-vehicle infotainment system development environment, students can conduct application development and testing, experiencing the complete intelligent connected vehicle software development process.

2.3.3 Industry integration

Linux's widespread application in the intelligent connected vehicle industry enables close alignment between academia and industry. Universities can collaborate with enterprises to introduce real project cases and development tools, conducting industry-academia-research cooperation projects. This enables students to access cutting-edge technologies and real application scenarios during their studies, improving their employment adaptability and professional quality.

3. Linux Teaching Methods

In the Intelligent Connected Vehicle major, teaching the Linux operating system is a critical component for cultivating students' practical abilities and problem-solving skills. To achieve optimal teaching effectiveness, the following methods can be employed.

3.1 Combining Theory with Practice

Linux operating system instruction cannot remain merely at the theoretical level; it must be closely integrated with practical application. Traditional instructional design employs a linear, theory-before-practice approach that schedules experimental sessions hours or even days after theoretical instruction. According to the Ebbinghaus forgetting curve, however, retention from passive listening drops below 40% after merely 20 minutes, and students' knowledge memory substantially decays within days, thereby missing the optimal learning window [6]. In the classroom, instructors should thoroughly explain fundamental Linux concepts, architecture, and common commands while simultaneously demonstrating operations to help students intuitively understand command functions and usage. For example, when teaching file operation commands, instructors can demonstrate on-site how to use `mkdir`, `cp`, `mv`, and other commands to create directory structures and files, then have students practice these operations in their own virtual machine environments. This theory-practice integration helps students better master the knowledge and enhances learning interest and efficiency.

Below is a specific teaching example demonstrating how instructors can use live demonstration combined with student hands-on practice to teach commands such as `mkdir`, `cp`, and `mv` for creating directory structures and files. This example can be utilized in practical sessions of Linux instruction within the Intelligent Connected Vehicle major.

3.1.1 Teaching objectives

(1) Students can proficiently use `mkdir` to create directories.

(2) Students can use `cp` to copy files and directories.

(3) Students can use `mv` to move or rename files and directories.

3.1.2 Teaching steps

Instructor Live Demonstration

(1) Open Terminal

The instructor first opens the terminal, demonstrating how to access the command-line interface.

(2) Create Directory Structure

The instructor demonstrates how to use `mkdir` to create multi-level directory structures. For example, creating a simulated Intelligent Connected Vehicle project directory structure:

```
mkdir -p ~/car_project/{src,docs,tests}
```

Explanation:

`mkdir` is the command to create directories.

The `-p` option enables recursive directory creation; if parent directories don't exist, they are created automatically.

`~/car_project/` is the main directory, and `{src,docs,tests}` are subdirectories created under the main directory.

(3) View Directory Structure

Use `ls` and `tree` commands to view the directory structure:

```
ls -l ~/car_project
tree ~/car_project
```

Explanation:

`ls -l` lists directory contents in detailed format.

The `tree` command displays directory structure in tree diagram format (if not installed, use `sudo apt install tree` to install it).

(4) Create File

Create a simple text file in the `src` directory:

```
echo "This is a sample file for the car project." > ~/car_project/src/sample.txt
```

Explanation:

`echo` is used to output strings.

`>` is used to redirect output to a file; if the file doesn't exist, it is created.

(5) Copy File

Use the `cp` command to copy a file from one directory to another:

```
cp ~/car_project/src/sample.txt ~/car_project/docs/
```

Explanation:

`cp` is the command to copy files.

The first path is the source file path, and the second is the destination path.

(6) Move or Rename File

Use the `mv` command to move or rename files:

```
mv ~/car_project/docs/sample.txt ~/car_project/docs/sample_document.txt
```

Explanation:

`mv` is the command to move or rename files.

If the destination path is an existing directory, the file will be moved into that directory.

If the destination path is a filename, the file will be renamed.

3.1.3 Student hands-on practice

(1) tudent Login to Virtual Machine

Students log into their own Linux virtual machine environment and open the terminal.

(2) Create the Same Directory Structure

Students follow the instructor's demonstration to create the same directory structure:

```
mkdir -p ~/car_project/{src,docs,tests}
```

(3) Create File

Create a simple text file in the `src` directory:

```
echo "This is my own sample file." > ~/car_project/src/my_sample.txt
```

(4) Copy File

Copy the file from the `src` directory to the `docs` directory:

```
cp ~/car_project/src/my_sample.txt ~/car_project/docs/
```

(5) Move or Rename File

Move the file from the `docs` directory to the `tests` directory and rename it:

```
mv ~/car_project/docs/my_sample.txt ~/car_project/tests/my_test_file.txt
```

(6) Verify Operations

Students use `ls` and `tree` commands to verify the directory structure and files are correct:

```
ls -l ~/car_project
tree ~/car_project
```

3.1.4 Teachers can guide students to summarize the following key points

(1) `mkdir` is used to create directories; the `-p` option can recursively create multi-level directories.

(2) `cp` is used to copy files or directories; source and destination paths must be specified.

(3) `mv` is used to move or rename files or directories; when moving, the destination path can be a directory, when renaming, the destination path should be a filename.

Through this combination of live demonstration and student hands-on practice, students can better understand and master these fundamental Linux commands, laying a solid foundation for subsequent Intelligent Connected Vehicle development environment setup and project practice.

3.2 Project-Driven Teaching Method

The project-driven teaching method is a teaching model oriented by practical projects. In Linux instruction, projects related to Intelligent Connected Vehicles can be designed, such as building an in-vehicle infotainment system development environment or developing simple in-vehicle applications. Through project practice, students can learn Linux operating system usage in real application scenarios and develop practical problem-solving abilities. During this process, students with strong hands-on abilities should assume a leadership role in assisting their less proficient peers, thereby fostering a spirit of teamwork [7]. For example, instructors can guide students to use Linux to build an Android Automotive OS-based in-vehicle infotainment system development environment, enabling them to learn how to configure development tools, write code, and debug programs within the project. The project-driven teaching method not only improves students' practical abilities but also cultivates their teamwork skills and innovative thinking.

3.3 Case Teaching Method

The case teaching method is a teaching approach that imparts knowledge and skills through analyzing and discussing actual cases. through case-based instruction and hands-on practice, students gain an in-depth understanding of the principles and methodologies of network service deployment, thereby enhancing their practical operational capabilities [8]. In Linux instruction, instructors can select actual cases related to Intelligent Connected Vehicles, such as autonomous driving system development or Internet of Vehicles device configuration, guiding students to analyze problems in cases and solve them using Linux commands and tools. For example, instructors can present a case of sensor data processing in an autonomous driving system, having students use Linux text processing tools (such as `grep`, `awk`, etc.) to analyze and process sensor data.
Below is a detailed step-by-step example of building an Android Automotive OS (AAOS) in-vehicle information system development environment based on Linux, for instructors to guide student learning and practice:
3.3.1 Environment preparation
(1) Hardware Requirements:
At least 16GB RAM.
At least 400GB available disk space.
High-performance processor recommended to accelerate compilation speed.
(2) Software Requirements:
Install Linux operating system; Ubuntu 18.04 or higher is recommended.
Install necessary software packages and tools, including Java Development Kit (JDK), Git, Repo, etc.
3.3.2 Installing basic tools
(1) Install OpenJDK 8:

```
sudo apt update
sudo apt install openjdk-8-jdk
```

Configure Java environment variables:

```
export
JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export PATH=$JAVA_HOME/bin:$PATH
```

Verify Java version:

```
java -version
```

(2) Install Git:

```
sudo apt install git
```

(3) Install Repo Tool:

```
mkdir ~/bin
curl
https://storage.googleapis.com/git-repo-downloads/repo > ~/bin/repo
chmod a+x ~/bin/repo
export PATH=~/bin:$PATH
```

(4) Install Other Dependencies:

```
sudo apt-get install git-core gnupg flex bison build-essential zip curl \
zlib1g-dev libc6-dev-i386 x11proto-core-dev libx11-dev \
lib32z1-dev libgl1-mesa-dev libxml2-utils xsltproc unzip fontconfig
```

3.3.3 Downloading android source code
(1) Create Working Directory:

```
mkdir ~/android-automotive
cd ~/android-automotive
```

(2) Initialize Repo:

```
repo init -u
https://android.googlesource.com/platform/manifest -b android-13.0.0_r83
```

(3) Sync Source Code:

```
repo sync
```

3.3.4 Configuring build environment
(1) Set Up Build Environment:

```
source build/envsetup.sh
```

(2) elect Build Target:

```
lunch aosp_car_x86_64-userdebug
```

This command configures the build target for automotive systems**.**
3.3.5 Compiling android automotive OS
(1) Start Compilation:

```
make -j$(nproc)
```

This command uses all available CPU cores for compilation; the process may take several hours.
(2) Resolve Compilation Issues:
If API update issues occur during compilation, run:

```
make update-api
```

3.3.6 Running the emulator
(1) Start Emulator:

```
emulator
```

If permission issues occur, run:

```
sudo chown $USER:$USER /dev/kvm
```

Then start the emulator again.

(2) Create Automotive-Specific AVD:
Open Android Studio.
Navigate to AVD Manager.
Create a new AVD using automotive system images.
Start the emulator.
3.3.7 Development and debugging
(1) Develop Application
Use Android Studio to create a new Android Automotive project.
Add necessary dependencies, for example:
implementation 'androidx.car.app:app-automotive:1.4.0-alpha01'
Create `CarAppService` and `Session` classes to implement application logic.
(2) Debug Application
Install the application on the emulator or device, and use the `adb` tool for debugging.
Through the above steps, students can successfully build an Android Automotive OS development environment in Linux and begin developing in-vehicle information system-related applications. Teachers can combine actual cases to guide students to gradually master development processes and techniques. Through case-based teaching, students can better understand Linux applications in the Intelligent Connected Vehicle field and improve their ability to solve practical problems.

4. Online Learning and Resource Sharing

Linux is an open-source operating system with abundant online learning resources and community support. Chinese MOOC platforms suitable for higher vocational colleges and widely disseminated include XuetangX and China University MOOC (iCourse), among others [9]. Teachers can guide students to utilize online learning platforms (such as Coursera, edX, etc.) to learn Linux-related courses and acquire the latest knowledge and technologies. Simultaneously, teachers can recommend open-source communities (such as GitHub, Stack Overflow, etc.) for students to participate in project development, ask questions, and exchange ideas, broadening their learning horizons. Rich online learning resources significantly enhance students' autonomy and flexibility in learning. By providing diversified, multi-tiered learning materials and interactive platforms, these resources not only facilitate the consolidation of classroom knowledge but also effectively broaden students' knowledge scope and stimulate their interest, thereby comprehensively improving overall learning effectiveness [10]. Through the comprehensive application of these teaching methods, the quality and effectiveness of Linux instruction can be effectively improved, cultivating high-quality talent with solid Linux skills and practical capabilities for the Intelligent Connected Vehicle major.

References

[1] Zhao, S., Xu, K., Song, J., & Wang, W. (2020). Analysis of intelligent connected vehicle operating system and its implementation strategy. Science and Technology Management Research, 2020(9), 107-111.
[2] Liu, X., Song, W., Cheng, Y., & Chen, Q. (Year). Exploration and practice of the application-oriented graduate cultivation model for intelligent connected vehicle under the background of "Four-Chain Integration". Automobile Applied Technology, *2025 (22)*, 104-109
[3] Jiang, F., Autonomous driving systems and street space reallocation: A paradigm shift from road redundancy to public space activation. Smart City, 2025(27),217-219
[4] Li, J. , Optimization of the professional curriculum system of intelligent networked vehicles based on the ability requirements of vocational positions. Automobile Education, 2025(24), 49-51
[5] Quan W., Research on the display application of virtual instrument based on Linux system and Qt development framework. Harbin: Harbin University of Science and Technology, 2017
[6] Yang, A., Zhang, Z., and Li, X., "Teaching reform and practice of Linux operating system course under the application-oriented talent training model," *Journal of Science of Teachers' College and University*, vol. 45, no. 11, pp. 80-83, 2025.
[7] Li, K., Construction of Embedded Operating System and Application Course Based on Linux, Automation Information Technology, no. 18, pp. 13-15, Sep 2025.
[8] Zhang, J., Zhu, K., Zhang, N., Wang, B., and Cao, L., A Study on the Reconstruction of Linux Operating System Management Curriculum System Based on Project-Based Teaching Method, Computer Knowledge

and Technology, vol. 21, no. 19, 172-175, July 2025.
[9] Song, W., & Qin, N. Exploration of curriculum reform for computer network major in higher vocational colleges based on blended teaching model—Taking "Linux System Services" course as an example. Journal of Guangxi Open University, 31(2), 2020.
[10]Wang, Y., & Wu, H. (2025). Exploration and practice of integrated curriculum teaching reform for "Fundamentals of Linux System" in the context of emerging engineering education. Journal of Hubei Open University, 45(2), 49-53.2025