

Development of an Integrated Platform for Local LLMs on Internal Servers

Shinya TAKUBO*, Ryusei IKEDA, Akinobu YOSHIMURA,
and Mitsuru HIRAKAWA

In recent years, with the rapid development of AI technologies, the utilization of AI in companies and research institutions has increased significantly. In particular, the expansion of Large Language Models (LLMs) for text generation has been prominent. To mitigate the risk of leaking confidential information, such as customer data and intellectual property (e.g., invention ideas), we are developing an integrated platform for local LLMs that operate on internal servers. In this environment, requests are sent from user PCs to internal servers, and inference is executed using computing resources on the internal servers, thereby realizing secure use of LLMs entirely confined within the company. Furthermore, we report that by sharing computing resources including GPUs, building client-based tools, and implementing custom application development environments via APIs, we have contributed to improving business efficiency and shortening research and development cycles while maintaining security.

Keywords: generative AI, local LLM

1. Introduction

1-1 Current status and challenges of AI utilization

In recent years, the utilization of AI in companies and research institutions has increased significantly with the rapid advancement of AI technology. Particularly notable is the expansion of large language models (LLMs), which are responsible for text generation. “cloud LLMs,” represented by OpenAI’s ChatGPT and Google’s Gemini, operate by sending user input data to servers managed by overseas vendors for computation, and then returning the results to the user. They have become integrated into society not only through chat systems but also via integration with diverse applications.

However, this high level of convenience comes with significant latent risks. From a corporate perspective, there are serious concerns regarding information leakage, as confidential information, internal documents, and customer data are transmitted externally. Furthermore, depending too heavily on specific vendors poses a business continuity risk should those vendors experience functional failures. While some organizations are attempting to reduce these risks by contracting with cloud providers to restrict data viewing and its use for AI training, the fundamental risk—that data is sent outside the company—cannot be completely eliminated as long as cloud LLMs are used. Indeed, the importance of this issue is strongly highlighted by the Information-technology Promotion Agency (IPA) in its Top 10 Information Security Threats,⁽¹⁾ where AI-related risks rose from being unranked to 3rd place in the 2026 edition.

Therefore, balancing the enhancement of corporate competitiveness through AI utilization with ensuring safety has become an urgent challenge.

1-2 Advancements in high-performance local LLMs

On the other hand, high-performance “Local LLMs,” which can be downloaded from the internet and operated in local environments such as a user’s PC, are being released one after another. These models possess recognition and generation capabilities for natural language, images, and audio that are comparable to those of cloud LLMs. Due to

rapid technological progress, the time gap for Local LLMs to catch up with the performance of the latest cloud LLMs is shortening. Figure 1 shows a comparison of the major benchmark performances between cloud LLMs and Local LLMs.

Furthermore, the development of platforms such as Hugging Face,⁽²⁾ where LLMs and data are made publicly available, is advancing. This has lowered the technical barriers to handling Local LLMs, making their utilization easier.

As high-performance LLMs that can run safely in local environments are becoming a reality, our company is working on building a “Local LLM Integration Platform” that allows us to utilize LLMs safely within our internal environment, thereby reducing our reliance on cloud LLMs.

In this paper, we will explain the internal server-based LLM platform in Chapter 2, the client-based LLM platform in Chapter 3, and the custom application development environment in Chapter 4.

2. Internal Server-Based LLM Platform

2-1 Server-based system configuration

As part of the verification process aimed at system implementation, a Local LLM tool was built on internal servers, enabling interactive chat and API integration via web access.

Figure 2 shows the overall system configuration of the server-based LLM platform.

Users access the local LLM tool via the web from their own PCs and send requests through a chat interface or via API. Requests received by the web server are allocated to internal GPUs on the servers for processing, and LLM inference results are returned to the user.

The main features of this method are as follows.

(1) Extensive LLM selection

LLMs differ in their characteristics depending on their architecture and training data. Therefore, this platform is equipped with more than 10 local LLMs and provides an interface that allows users to freely choose among them. Furthermore, it is possible to send a single request simultaneously to multiple LLMs and display each response side by side. This enables users to efficiently compare multiple results and extract the optimal answer from them.

(2) User authentication and LLM management

To prevent unauthorized use by unregistered users, we have implemented user authentication functions. This not only ensures proper management of user information but also allows for the customization and deployment of LLMs tailored to the operational characteristics and usage scenarios of each department, thereby promoting safe utili-

zation within the company.

(3) Computing resources

This platform is not built on a single GPU server but is distributed across multiple servers. It optimally allocates resources and assigns workloads to the most suitable servers based on the parameter scale and memory requirements of the hosted LLMs, as well as the specifications of each GPU, to maximize inference performance.

2-2 Management method of computing resources

Computational resources during LLM inference are managed using a centralized management system.

This is a management approach that dynamically shares GPU resources among multiple applications, maximizing resource utilization while reducing the burden of environment setup for developers.

The main advantages of this method are as follows.

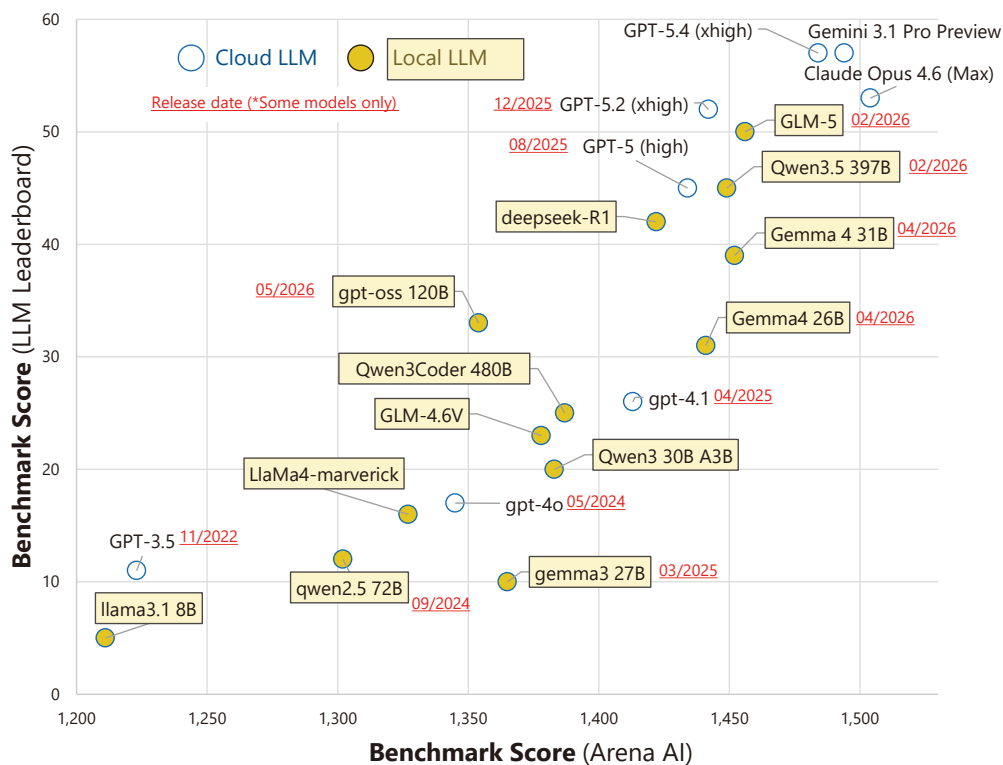


Fig. 1. Performance comparison of cloud LLM and local LLM (as of April 2026)
(Comparison on major benchmarks: LLM Leaderboard⁽³⁾ and Arena AI⁽⁴⁾)

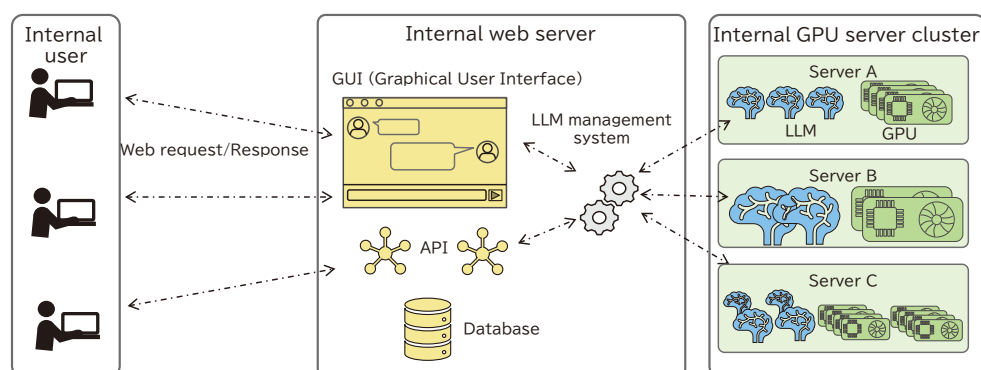


Fig. 2. Overall architecture of the internal server-based LLM platform

(1) Resource sharing with other applications

In addition to LLMs, our company has developed and shared within the organization a wide range of applications requiring GPU-accelerated computation, such as AI tools for image recognition and tools for scientific computing simulations. Therefore, instead of fixing GPU allocation for each application, we dynamically allocate GPUs in response to requests. This approach prevents the inefficiencies and increased costs associated with assigning GPUs for single-purpose use, thereby maximizing overall resource utilization.

(2) Advantages of sharing computational environments

The technical staff in our company's business units and research institutes, who we consider to be our users, have very high programming skills. However, since they are not experts in information processing, building advanced internal computing environments has been difficult and labor-intensive for them. This process involves complex tasks, such as approving expensive servers, ensuring power supply, managing software versions, and adjusting dependent libraries. Therefore, this system aims to centralize the maintenance of GPUs and the infrastructure of inference servers, allowing each technical expert to focus on utilizing various applications, including generative AI, while reducing the effort required for environment setup.

2-3 Improvements obtained through verification

After launching an internal server-based LLM platform with the above-mentioned features for trial to hundreds of users, daily usage reached tens of thousands of requests.

At the same time, the following points have emerged as new requests for the system.

(1) Seamless collaboration with personal knowledge

To expand the business use of LLMs, it is essential to have a mechanism that allows for referencing and reusing past development history and individuals' accumulated knowledge. Therefore, we will build an environment where user personal data handled daily and LLMs are more closely integrated to maximize their utilization.

(2) Provision of a custom application development environment

In the server-based configuration shown in Fig. 2, the development environment was limited to the system side, making it difficult for users to implement and verify func-

tions themselves.

Therefore, we decided to proceed with building a flexible application optimized for each user, alongside conducting benchmark tests of general-purpose knowledge management tools such as Obsidian,⁽⁵⁾ and to establish an environment that allows for rapid debugging and iterative improvement on the users' own PCs. This approach makes it possible to maintain the convenience of server-based systems while maximizing the flexibility of individual data utilization and application development.

Based on the verification results above, we have constructed a "Client-based LLM platform" that installs and utilizes applications on business PCs, which will be explained in the next chapter.

3. Client-based LLM Platform

3-1 Client-based system configuration

Figure 3 shows the overall system configuration of a client-based LLM platform.

First, the user installs a lightweight client application on their own PC. The application provides tools for access via the web, similar to the server-based version. Subsequently, like the server-based version, it is configured to send requests from the chat interface or via the API. The computing resources used during LLM inference maintain inference speed by adopting the centralized management method shown in Section 2-2.

3-2 Advantages of the system

The greatest advantage of this system is its seamless integration with data on the user's PC.

For example, by simply specifying a folder path on a company server to which the user has access, the LLM can perform investigations and analysis on all documents within the folder.

Additionally, by providing a debugging environment simultaneously, users will be able to install flexible applications tailored to specific departments or users within their tools.

Additionally, since conversation history and attached files are distributed and saved only on your business PC, you can view the conversation history stored on the PC even if the network connection is temporarily lost, and it will automatically sync upon reconnection.

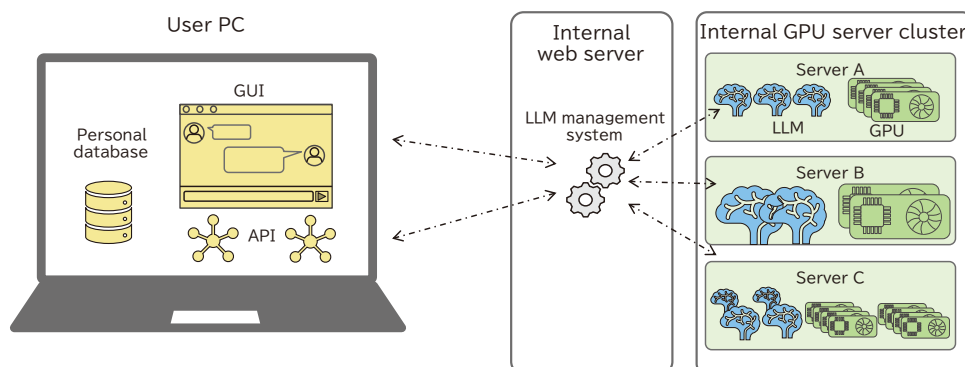


Fig. 3. Overall configuration of the client-based LLM platform

Table 1 compares the features of the server-type and client-type described above.

Table 1. Comparison of server and client types

Features	Server-based	Client-based
Execution environment	Internal server	User PC
File collaboration	Not good (Attach required files)	Good (Specify the reference folder)
Debugging environment	None	Available
Introduction cost	Low (No configuration required)	Medium (Installation required)
Computational resources	Centralized management	Centralized management

Users can choose between server-type and client-type configurations based on use cases such as the need for file collaboration and the availability of a debug environment.

3-3 Providing application development environments

Client-based LLM platforms do not merely provide a chat interface; they also simultaneously offer an environment where users can develop and deploy custom applications.

The client applications on this platform come standard with an API library for integration with LLMs and a development toolkit. This enables users to freely develop and deploy applications optimized for their own personal or department-specific workflows on their own PCs. Figure 4 shows an overview of the application development environment.

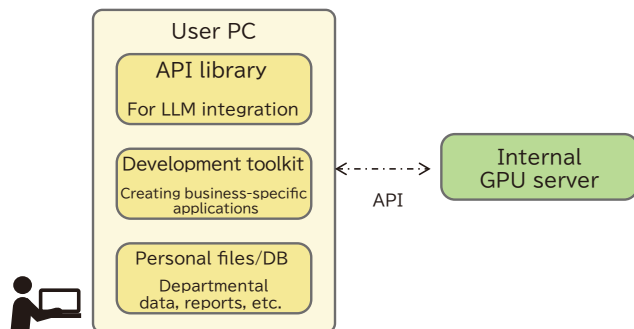


Fig. 4. Overview of the application development environment

The development environment is designed to be simple, allowing practical tools to be built with just a few dozen lines of code in Python. For example, by combining multiple LLMs, one can easily implement processes on their own PC where the first LLM reads all meeting minutes from a personal folder, the results are passed to a second LLM to convert them into a specific report format, and then an email is sent to internal employees (Fig. 5).

Note that, as shown in the configuration in Fig. 3, the inference process itself utilizes the GPU resources of the internal server, enabling fast response times without depending on the performance of the business PC.

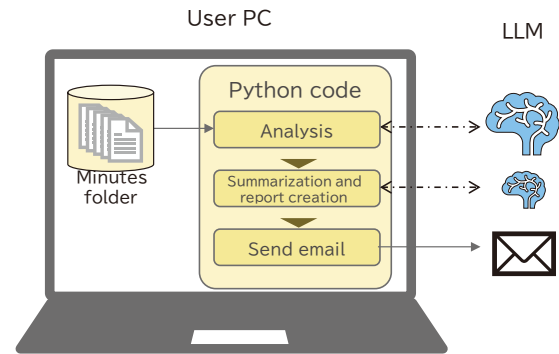


Fig. 5. Example of a custom application

3-4 Mechanism of the application sharing hub

To enable the organization-wide use of individually developed applications, this platform includes an “App Registration Hub” feature. If a user wants to deploy a developed application to other users, they register the application in the hub. Registered applications are then published on the platform after undergoing security and operational verification by administrators.

Other users can easily install the required applications into their own PC environments by searching for and downloading them from the hub. Figure 6 shows the flow from application development to sharing and deployment.

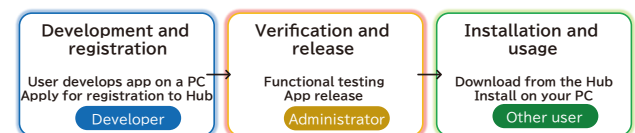


Fig. 6. Application development and sharing flow

This mechanism facilitates the horizontal deployment of superior applications, accelerating business efficiency across the entire organization. Additionally, since application version management is centrally handled on the hub, users can always access the latest and most stable versions.

4. Conclusion

To achieve a balance between enhancing corporate competitiveness and ensuring security, we built a “Local LLM Integration Platform” that operates on internal servers. By implementing centralized management and sharing of GPU computing resources, we reduced the burden of environment setup for developers while maximizing the utilization of limited computing resources. Furthermore, the introduction of a client-based platform enabled seamless integration with local files on business PCs and provided debug environments optimized for individual users, thereby lowering the barrier to LLM adoption. Additionally, by establishing a custom application develop-

ment environment and an app-sharing hub, we made it possible to widely deploy tools customized to the specific workflows of individual departments or individuals.

Currently, this platform has been deployed across multiple departments within the company, accelerating the practical application of LLMs. It is crucial to verify diverse use cases and identify high-value applications and systems for business operations. In the future, we will collaborate with the Information Systems Department to advance the implementation and deployment of this platform as a standard internal foundation. By doing so, we aim to eliminate the risk of confidential information leaking outside the organization while achieving operational efficiency and shortening the R&D cycle, thereby establishing sustainable competitive advantages.

- ChatGPT is a trademark or registered trademark of OpenAI, Inc. or its subsidiaries in the U.S. and other countries.
- Gemini is a trademark or registered trademark of Google Inc. in the U.S. and other countries.
- Hugging Face is a trademark or registered trademark of Hugging Face, Inc. in the U.S. and other countries.
- GPT and GPT-4 are trademarks or registered trademarks of OpenAI, Inc. or its subsidiaries in the U.S. and other countries.
- Claude is a trademark or registered trademark of Anthropic, PBC in the U.S. and other countries.
- llama and LLaMa are trademarks or registered trademarks of Meta in the U.S. and other countries.
- deepseek is a trademark or registered trademark of 杭州深度求索人工智能 基礎技術研究有限公司 .
- Qwen is a trademark or registered trademark of ALIBABA INNOVATION PRIVATE LIMITED.
- Gemma is a trademark or registered trademark of Google Inc. in the U.S. and other countries.
- Obsidian is a trademark or registered trademark of Dynalist Inc.

References

- (1) BrainDirection Inc., "Security Trends 2026: IPA Top 10 Threats and Gartner Outlook"
<https://www.brain-d.jp/en/tech-security-trends-2026.html> (accessed 2026-03-16)
- (2) Hugging Face, "The AI community building the future"
<https://huggingface.co/> (accessed 2026-03-16)
- (3) Artificial Analysis, "LLM Leaderboard- Comparison of AI models from OpenAI, Anthropic, Google, SpaceXAI & others"
<https://artificialanalysis.ai/leaderboards/models> (accessed 2026-04-01)
- (4) Arena Intelligence Inc., "Overview"
<https://arena.ai/leaderboard> (accessed 2026-04-01)
- (5) Obsidian, "Sharpen your thinking"
<https://obsidian.md/> (accessed 2026-03-31)

Contributors

The lead author is indicated by an asterisk (*).

S. TAKUBO*

• Principal Engineer, Digital Transformation Laboratory



R. IKEDA

• Digital Transformation Laboratory



A. YOSHIMURA

• Group Manager, Digital Transformation Laboratory



M. HIRAKAWA

• Director, Digital Transformation Laboratory

