A Study on Design Requirements for Ensuring Cross-Compatibility in Korean Public-Sector AI Systems

NunSol Park
NIA(National Information Society Agency)
Daegu, Republic of Korea
nspark@nia.or.kr

Eunju Kim
NIA(National Information Society Agency)
Daegu, Republic of Korea
outframe@nia.or.kr

DongHee Kim
NIA(National Information Society Agency)
Daegu, Republic of Korea
dhk@nia.or.kr

HanKyoung Choi
NIA(National Information Society Agency
Daegu, Republic of Korea
hotspot@nia.or.kr

JinKyu Yang
NIA(National Information Society Agency)
Daegu, Republic of Korea
jky@nia.or.kr

Abstract*—With the rapid advancement of artificial intelligence (AI) technologies and the diversification of commercial models, the demand for AI adoption in the public sector has surged dramatically. Governments and public institutions are actively leveraging AI technologies to enhance administrative efficiency, deliver personalized services, and enable data-driven policy-making. However, in the actual implementation process, numerous limitations and challenges have emerged. In particular, when AI systems are designed in a way that is dependent on specific cloud platforms or proprietary models, organizations are often burdened with high costs and technical constraints during system transitions, expansions, and maintenance in response to evolving technologies. In such a fast-changing AI ecosystem, ensuring the stability and scalability of AI services in the public sector requires establishing cross-compatibility across AI infrastructure, cloud platforms, and AI models. Cross-compatibility refers to the ability for services, data, and models to be seamlessly interoperable and reusable across different AI environments. This enables public institutions to avoid vendor lock-in and adopt a diverse range of suppliers and solutions. Moreover, securing cross-compatibility acts as a strategic means to reduce long-term maintenance costs while strengthening the sustainability and technological autonomy of public AI services. Nevertheless, many current AI projects in the Korean public sector are being implemented based on heterogeneous standards, model formats, and platform environments. As a result, compatibility issues frequently arise during model replacement or technology migration processes. To effectively introduce and manage diverse AI technologies, it is imperative for public institutions to adopt cross-compatibility strategies grounded in platform independence and open standards. Therefore, this paper proposes design-level requirements aimed at securing cross-compatibility in AI services for the public sector.

***Keywords*—Cross-Compatibility, Korean public AI system, AI service, Cloud platform, AI model**

I. INTRODUCTION

A. Current Status of Public AI Adoption in South Korea

In recent years, the South Korean government and public institutions have been actively promoting the adoption of artificial intelligence (AI)–based services to address various societal challenges and enhance administrative efficiency [1], [2]. Numerous applications are emerging, including automated civil response systems, intelligent document analysis, and predictive analytics for policy and service delivery [3]. These initiatives are being implemented not only at the central government level but also across local governments and public agencies, leading to a broad diffusion of AI pilot projects throughout the public sector. Despite this rapid expansion, many AI projects are developed in a vendor-dependent manner, relying heavily on specific platforms or technologies [4], [5]. As a result, these systems often face significant limitations when undergoing technological upgrades, expansions, or transitions. For instance, AI models built on proprietary frameworks—such as "PyTorch" or "TensorFlow"—or stored in custom formats, pose major challenges in terms of reusability or migration to other environments [6]. To overcome these issues, it is essential to ensure cross-compatibility among AI infrastructures, platforms, and models. This will allow public institutions to remain technologically agile, reduce operational and maintenance costs, and avoid vendor lock-in, thereby securing sustainable and scalable AI services [7].

B. Definition of Cross-Compatibility

Cross-compatibility refers to the characteristic that enables AI models to freely interconnect, transfer, and migrate functions and data across different platforms, systems, and infrastructures [8] without technical constraints, thereby allowing the implementation of flexible AI service environments that are not dependent on specific technologies. The detailed components of cross-compatibility can be divided into two categories: the first is cross-compatibility with respect to changes in AI infrastructure, and the second is cross-compatibility with respect to changes in AI models. Cross-compatibility with respect to AI infrastructure changes refers to the ability of AI models to be deployed and operate without additional service development even when the AI infrastructure environment—such as cloud platforms, servers, or GPUs—changes [9]. Cross-compatibility with respect to AI model changes refers to the ability to flexibly replace and update various AI models while maintaining the existing system architecture and API formats [10].

II. Structural Considerations for Ensuring Cross-Compatibility in Public AI

First, the requirements for ensuring cross-compatibility in response to changes in AI infrastructure are as follows. Public AI services should be able to operate continuously without requiring additional redevelopment, even when underlying infrastructure environments—such as cloud platforms, server configurations, or GPU resources—change [11]. To achieve this, AI models must be stored and deployed using container-based environments [12]. Container technologies, such as Docker, package all components required for AI model execution—including source code, libraries, framework versions, and runtime configurations—into a single, self-contained unit [13]. This approach effectively mitigates compatibility issues that may arise from differences across operating environments. In particular, public-sector AI systems are highly likely to be deployed across diverse infrastructure environments, including on-premises systems, private clouds, public clouds, and hybrid cloud architectures. Consequently, deployment mechanisms that are independent of specific servers or operating systems are essential. Container-based deployment satisfies these requirements by enabling the same container image to be consistently applied across heterogeneous infrastructure environments, thereby ensuring both portability and reproducibility of AI models [14].

Furthermore, container images support version control, which facilitates efficient model updates and rollbacks, while also enhancing responsiveness to operational failures. From the perspective of long-term public AI service operation, this contributes to achieving both system stability and improved maintenance efficiency. In addition, manual, operation-intensive deployment processes increase the likelihood of errors when infrastructure environments change. Therefore, the establishment of automated deployment pipelines using Continuous Integration and Continuous Deployment (CI/CD) tools is required. By leveraging tools such as GitLab CI and Jenkins to automate the processes of container image building, testing, and deployment, AI models can be deployed and operated in a consistent manner regardless of changes in the infrastructure environment. Such automated deployment pipelines play a critical role in ensuring operational consistency in public AI systems and in minimizing technical burdens associated with infrastructure expansion or migration.

In conclusion, by jointly adopting container-based storage and deployment mechanisms along with CI/CD automation environments, public AI systems can respond flexibly to infrastructure changes and secure cross-compatibility that is independent of specific cloud platforms or hardware environments. Furthermore, establishing a containerized infrastructure with CI/CD automation enables public AI systems to remain resilient and interoperable, regardless of the underlying cloud services or hardware environments.

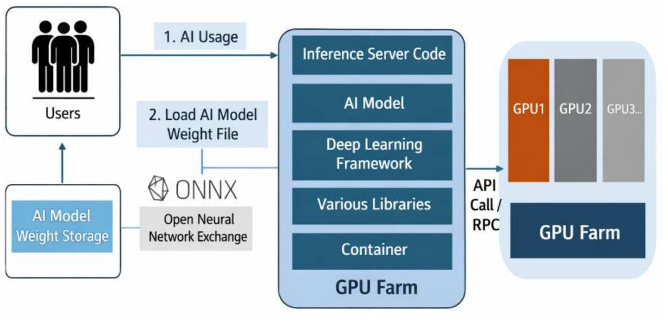

Fig. 1. Example of Container-Based AI Model Execution

A. Containerization

- A key strategy for deploying AI models into operational environments and ensuring their portability is containerization. Containers encapsulate all components required for AI model execution—including source code, libraries, and configuration files—into a single, self-contained package, enabling consistent deployment across heterogeneous environments. Docker-based containers are commonly employed for this purpose, and the following procedures are required.

- Container Image Construction: A container image is built using a “Dockerfile” by encapsulating the trained AI model together with execution scripts, required libraries, and configuration files into a single image.

- Lightweight Design and Optimization: Since public-sector environments often impose constraints on cloud infrastructure resource usage, the container image size is minimized through model size optimization and the removal of unnecessary dependencies[15].

- Compatibility Testing: The configured container is validated to ensure that it can be executed without issues across various platform or cloud environments, such as Kubernetes, OpenShift, Naver Cloud Platform, and NHN Cloud.

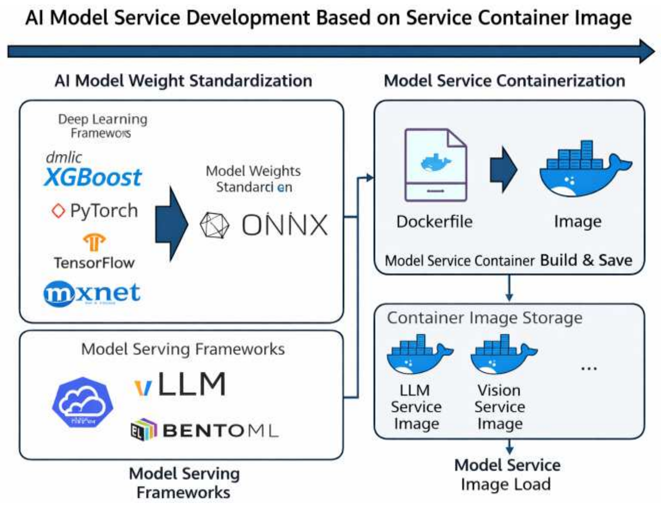

Fig. 2. Example of Service Container Image of AI Model Service

One of the most critical factors in replacing or updating AI models for improved performance is the reusability of training data. To enable this, it is essential to standardize the structure and format of training data in advance. In the public sector, AI services are not limited to a single model but must evolve continuously by adopting and applying various algorithms and models as technologies advance. If the data structure is tightly coupled with specific models, every model change would require reprocessing or restructuring of data, leading to inefficiency.

Specifically, the training and test datasets used for AI model development must have a consistently defined structure, including feature names, data types, units, handling of missing values, and normalization standards. Without such structural consistency, even if datasets remain the same, each model may require a different preprocessing process. This not only undermines the comparability of training outcomes but also reduces the reliability of performance verification. Therefore, defining data schemas in advance and managing feature names and data types in a standardized manner is essential.

Furthermore, consistency in label definition is also crucial, especially in tasks such as classification, regression, and prediction. For example, if class definitions in a multi-class classification task vary across models or if label encoding methods differ, existing datasets may not be reusable for training new models[16]. To address this, the semantics, range, and encoding rules of labels must be clearly documented and consistently maintained, ensuring identical learning conditions during model replacement. Beyond the model training phase, data standardization plays a pivotal role in the full lifecycle management of AI systems. This includes model performance comparison, retraining, performance degradation detection, auditability, and interpretability. In the public sector, where transparency and accountability in AI decision-making are crucial, standardized data structures and label definitions help minimize policy and administrative risks associated with model changes.

Ultimately, standardizing the structure and format of training data is a prerequisite for ensuring cross-compatibility during AI model replacement and updates. This enables public institutions to continuously leverage their existing data assets while flexibly adopting the latest AI models, laying the foundation for sustainable and adaptive AI service deployment.

TABLE I. RECOMMENDED DATA FORMATS FOR DATA REUSABILITY

Format	Coverage	Explanation
JSONL (JSON Lines)	**Targeting Reusability Across NLP, LLM, and Multimodal Models**	**- Each row is an independent JSON object - Enables complex label structures such as text and multi-label formats**
CSV	**Other General Fine-Tuning**	**- Universal file format - Applicable to all types of models**

It is essential to standardize the interface between AI models and information systems. Given the rapid evolution in the architecture and scale of AI models, input/output (I/O) interfaces must be designed in a forward-compatible manner to accommodate future model version updates. This includes preparing for changes in model outputs and ensuring compatibility with existing system structures. In particular, systems should implement mechanisms to compare outputs between the existing and updated models to evaluate compatibility and impact. When significant changes occur, a separate versioning and management strategy for the API should be maintained to support stable integration.

Furthermore, to enable process reuse, lifecycle metadata must be systematically documented and managed. This involves capturing metadata across all components of the AI development pipeline, including data preprocessing scripts, model architectures, and configuration parameters. In addition, metadata should describe the training environment, such as the frameworks used and library versions. Establishing a standardized training environment based on consistent data structures and metadata ensures the reproducibility of results and maintains consistent model quality and operational stability during model replacement or retraining.

B. Standardization of Input/Output Interfaces

- Designing Interfaces with Version Updates in Mind: As AI models are frequently updated due to performance improvements or changes in algorithms, it is necessary to ensure that input/output (I/O) interfaces are designed with future version updates in mind. While the input structure should ideally remain consistent, the output format may vary depending on the model version. Therefore, the interface should be designed to ensure backward compatibility, allowing newer models to function seamlessly within existing systems without disrupting downstream processes.

- Introducing an API Versioning System: n cases where significant changes to the interface are required, a well-defined versioning system must be introduced to avoid conflicts with existing APIs. For instance, maintaining separate versions such as /v1/predict and /v2/predict allows parallel operation of multiple API versions, ensuring stable integration with legacy systems while accommodating new capabilities.

- In American English, commas, semicolons, periods, question and exclamation marks are located within quotation marks only when a complete thought or name is cited, such as a title or full quotation. When quotation marks are used, instead of a bold or italic typeface, to highlight a word or phrase, punctuation should appear outside of the quotation marks. A parenthetical phrase or statement at the end of a sentence is punctuated outside of the closing parenthesis (like this). (A parenthetical sentence is punctuated within the parentheses.)

- Implementation of Output Comparison Mechanisms: By implementing functionalities that enable comparative analysis of prediction results between the existing model and the updated model—such as visualization of output differences or generation of comparative reports on key indicators—it becomes possible to quantitatively assess the impact of model updates on policy application and service outcomes[17].

C. Full-Cycle Metadata Documentation and Managament

- Metadata Structuring of Training Pipeline Components: It is essential to systematically organize metadata for all components used throughout the entire process of data collection, preprocessing, training, and model deployment. This metadata should include the following elements(Parameters, Model architecture, Training logs and performance metrics, etc.).
- Documentation of Training Environment Information: To ensure reproducibility and consistent model quality, detailed information about the environment in which the model was trained must also be included as metadata such as Frameworks & versions(e.g., PyTorch 2.0, TensorFlow 2.11), Hardware specifications(e.g., NVIDIA A100, CUDA 11.7) and libraries and versions.

III. IMPLICATION

This study derives the core requirements for ensuring cross-compatibility to enable the stable adoption and widespread deployment of artificial intelligence (AI) systems in the public sector. As a result, it is expected to yield the following practical implications across future government- and public-sector AI initiatives.

A. Reduction of Technology Dependency and Strengthening of Platform Independence

When public institutions adopt AI services, technical dependency on specific vendors, cloud platforms, or frameworks often leads to various challenges, including increased long-term maintenance costs, reduced operational flexibility, and limited ability to respond to rapid technological evolution. The container-based deployment approaches, standardized model formats, and API interface standardization proposed in this study mitigate such technology dependency by enabling AI models to be reused and migrated across diverse environments. Consequently, these measures enhance platform independence and technological autonomy, providing a foundation for strengthening governmental digital sovereignty.

B. Improvement of Operational Efficiency and Reduction of Maintenance Costs

Following the deployment of AI systems, continuous operation, performance maintenance, and periodic model replacement are required. If critical elements such as data formats, input/output structures, and training environments are not managed consistently, substantial time and cost are incurred whenever new models are introduced. The standardization of data and metadata, along with the metadata-based documentation of training pipelines proposed in this study, minimizes redundant work and ensures consistent quality during model replacement or retraining. This approach improves maintenance efficiency while also enhancing transparency and predictability in budget execution.

C. Ensuring Scalability and Sustainability of AI Services

AI technologies are evolving rapidly, and the functional requirements of public services are becoming increasingly complex. Accordingly, AI systems must be continuously improved and expanded to incorporate diverse models, including multimodal models, large language models (LLMs), and prediction-based models. This study proposes a strategy that structurally embeds cross-compatibility to proactively respond to such technological advancements. As a result, it establishes a foundation for implementing sustainable AI services that are not constrained by specific technological environments.

D. Establishment of a National-Level AI System Foundation

Finally, the design requirements for ensuring cross-compatibility extend beyond individual institutional projects and serve as a foundation for standardizing and expanding technical infrastructures at the national AI ecosystem level. As a foundational study that considers future AI technical standards, public data standardization, and institutionalization through collaboration with organizations such as TTA and other standardization bodies, this research is expected to provide substantial momentum toward the realization of a digital platform government.

IV. CONCLUSION

This study proposed a set of design-level requirements for ensuring cross-compatibility in public-sector AI systems as a strategic foundation for sustainable operation and technical autonomy. Focusing on two primary axes—changes in AI infrastructure and changes in AI models—we identified and elaborated on key technical considerations and implementation strategies.

First, in response to infrastructure shifts, we emphasized the importance of container-based deployment and storing AI models in standardized formats (e.g., ONNX, PMML) to enhance portability and scalability. Second, to facilitate the replacement or upgrading of AI models, we discussed the necessity of standardizing data structures and label definitions, managing API versioning, implementing output comparison mechanisms, and maintaining lifecycle metadata in a systematic manner.

The proposed framework aims to help public institutions build a flexible AI ecosystem that is not locked into specific vendors or technologies, thereby enabling the integration and operation of diverse AI solutions. Moreover, the requirements presented here can serve as foundational criteria for public agencies in AI project planning, RFP documentation, technical verification, and performance evaluation. With the accumulation of real-world implementation cases, this framework can further evolve into a robust set of technical and policy guidelines.

For future research, it will be important to validate the practical applicability of these design requirements by constructing an empirical implementation framework, as well as to develop more refined standards through comparative analysis of interoperability practices across countries and industries. These efforts will contribute to the establishment of a more robust, scalable, and policy-aligned foundation for future AI adoption in the public sector.

References

[1] OECD, AI in the Public Sector: Opportunities and Challenges, OECD Publishing, Paris, 2021.

[2] European Commission, White Paper on Artificial Intelligence: A European Approach to Excellence and Trust, Brussels, Belgium, 2020.

[3] M. A. Cusumano, A. Gawer, and D. B. Yoffie, The Business of Platforms, Harper Business, New York, NY, USA, 2019.

[4] NIST, AI Risk Management Framework (AI RMF 1.0), National Institute of Standards and Technology, Gaithersburg, MD, USA, 2023.

[5] J. Kreps et al., “Managing Machine Learning Models at Scale,” in Proc. IEEE Int. Conf. Big Data, 2019, pp. 110–117.

[6] European Union, Interoperable Europe Act, European Commission, Brussels, Belgium, 2023.

[7] ISO/IEC 19941, Information Technology — Cloud Computing — Interoperability and Portability, ISO, Geneva, Switzerland, 2017.

[8] P. Mell and T. Grance, The NIST Definition of Cloud Computing, NIST SP 800-145, 2011.

[9] M. Zaharia et al., “MLflow: An Open Platform for the Machine Learning Lifecycle,” Proc. VLDB Endowment, vol. 13, no. 12, pp. 3459–3462, 2020.

[10] NIST, AI Risk Management Framework (AI RMF 1.0), National Institute of Standards and Technology, Gaithersburg, MD, USA, 2023.

[11] Docker Inc., Docker Documentation: Overview of Docker Containers, https://docs.docker.com/get-started/, accessed 2025.

[12] P. Pahl, “Containerization and the PaaS Cloud,” IEEE Cloud Computing, vol. 2, no. 3, pp. 24–31, May–June 2015.

[13] Kubernetes, Kubernetes Documentation, https://kubernetes.io/docs/, accessed 2025.

[14] M. Villamizar et al., “Evaluating the Monolithic and the Microservice Architecture Pattern,” IEEE Software, vol. 32, no. 3, pp. 42–50, 2015.

[15] J. Merkel, “Docker: Lightweight Linux Containers for Consistent Development and Deployment,” Linux Journal, no. 239, 2014.

[16] NIST, Big Data Interoperability Framework: Volume 3, Use Cases and General Requirements, NIST Special Publication 1500-3, 2015.

[17] NIST, Big Data Interoperability Framework: Volume 3, Use Cases and General Requirements, NIST Special Publication 1500-3, 2015.

.