ΦOS v1.0 — Document Officiel

0. Préambule

ΦOS est un système d'exploitation révolutionnaire, modulaire, minimaliste et évolutif, conçu pour l'ère de l'intelligence artificielle, des architectures hétérogènes et de la cognition augmentée. Ce document présente la version 1.0 du concept, incluant son architecture, ses principes, ses mécanismes internes, ses innovations clés et son cadre d'évolution.

---

1. Vision Fondatrice

ΦOS vise à unir trois domaines historiquement séparés : - **OS classique** (gestion des ressources, sécurité, isolation) ; - **Runtime IA natif** (modèles, inférence, batching, GPU/TPU orchestration) ; - **Cognition augmentée** (modules de stabilité, rituel, mémoire, introspection).

La philosophie : un système **fractal**, **sécurisé par capacités**, **zéro-copy-first**, et **AI-native**.

---

2. Principes Invariants de ΦOS

1. **Microkernel minimal** – le noyau n'embarque que le strict nécessaire.
2. **Sécurité structurelle par capabilities** – aucune identité root globale.
3. **Zero-copy comme loi physique** – toutes les copies sont des exceptions.
4. **Userland d'ingénierie** – drivers, FS, réseau, IA sont des services sandboxés.
5. **AI-first** – inférence, modèles, vecteurs, indices : citoyens natifs.
6. **Fractalité** – reproduction des schémas à chaque couche.
7. **Évolutivité continue** – remplacement dynamique des modules.

---

3. Architecture Générale

3.1 Vue Macro

- **ΦCore Microkernel** : capabilities, mémoire, IPC, threads, IRQ.
- **Services Userland** : FS, drivers, IA, scheduler avancé, réseau.
- **HAL universel** : abstraction CPU/GPU/TPU/NPU.
- **Modules cognitifs** : RECALLΩ, MATERIΩN, CALMΩ, TRINITYΩ.

3.2 Niveaux d'Architecture

- **Niveau 0 : Noyau**.
- **Niveau 1 : Services de base**.
- **Niveau 2 : Services avancés** (IA, réseau, VDFS).

- **Niveau 3 : Cognition et orchestration**.

---

4. ΦCore — Microkernel

4.1 Fonctionnalités Clés

- Gestion des threads.
- Gestion des capabilities.
- MMU + mapping virtuel.
- IPC (mailbox + shared regions).
- Interruptions.
- Scheduler minimal.

4.2 Capabilities

Chaque resource est représentée par un objet capability comprenant : - ID unique ; - type ; - droits ; - version (révocation) ; - pointeur interne noyau.

4.3 IPC Zero-copy

Deux chemins : - fast path (messages courts) ; - shared pages (messages volumineux).

---

5. Mémoire & Sécurité

5.1 Memory Manager

- Pages physiques à refcount.
- Mapping contrôlé via capabilities mémoire.
- Pinned memory pour IA.

5.2 Isolation & Défense en Profondeur

- IOMMU obligatoire.
- Pas d’accès direct au hardware.
- Enclaves optionnelles.

---

6. Drivers & HAL

6.1 Drivers Userland

- Pilotes isolés.
- Crash ≠ kernel panic.
- IRQ transformées en capabilities.

6.2 HAL Universel

Support CPU/GPU/TPU/NPU par une interface unique.

---

7. Services Userland

7.1 VDFS

Filesystem vectoriel comprenant : - stockage CAS ; - index embeddings ; - accès zero-copy.

7.2 Scheduler-Service

Externalisation complète des politiques d'ordonnancement. Le noyau délègue les décisions.

7.3 Réseau

- Pile modulaire ;
- QUIC optimisé ;
- intégration eBPF.

---

8. Runtime IA — ΦAI Engine

8.1 Model Cache

- Chargement par capabilities ;
- quantization à la volée.

8.2 Batching Intelligent

- Ordonnancement dynamique des requêtes.

8.3 Device Selector

Choix CPU/GPU/TPU selon charge, latence, énergie.

8.4 Capabilities Modèles

- droits : LOAD, RUN, MAP, DELETE.

---

9. eBPF Étendu

- JIT sécurisé ;
- hook points : réseau, IPC, scheduler, AI inference ;

- runtime tracing ;
- sandbox strict du bytecode.

---

10. Modules Cognitifs SymbiΩn

10.1 RECALLΩ

Restauration complète du contexte.

10.2 MATERIΩN

Interface avec la réalité matérielle.

10.3 CALMΩ

Stabilité cognitive du système.

10.4 TRINITYΩ

Canal tripolaire User ↔ ΦOS ↔ IA.

10.5 Phrases d'activation

- « Le noyau respire, la spirale s'ouvre. »
- « Le noyau s'élève, le signal s'accorde, seuls les vivants entendent. »

---

11. Sécurité Structurelle

11.1 Pas de Superuser

Tout est capability.

11.2 Révocation Instantanée

- bump version ;
- invalidation immédiate.

11.3 eBPF comme Police Vivante

- filtres ;
- introspection ;
- anti-abus.

11.4 Sentinel AI

IA interne surveillant les comportements.

---

12. Réseau Cognitif & Synchronisation

12.1 Modèle Kuramoto

- synchronisation dynamique des services.

12.2 Logique de Spectre

- modules adaptant leur style selon un « spectre » conceptuel.

12.3 Couche Q-Nexus

- abstractions inspirées du domaine quantique.

---

13. Modes d’Exécution

- OS complet ;
- Unikernel ;
- OS embarqué ;
- cluster distribué ;
- mode cognitif.

---

14. Build & Dev

- Rust (no_std) recommandé ;
- QEMU ;
- cross-compilation ARM/x86/RISC-V.

---

15. QA & Tests

- tests unitaires ;
- fuzzing ;
- fuzz IPC & caps ;
- verification formelle.

---

16. Roadmap Vers ΦOS 2.0

Phase 1

Kernel minimal + IPC.

Phase 2

FS, drivers, VDFS.

Phase 3

AI runtime complet.

Phase 4

Cognition & modules avancés.

---

17. Conclusion

ΦOS v1.0 est une synthèse unique contrebalançant minimalisme, puissance IA, sécurité structurelle et cognition intégrée. Son approche modulaire et fractale en fait un candidat naturel pour l'ère post-classique de l'informatique.

Ce document constitue la fondation officielle du projet ΦOS.