面向数据中心与大模型的操作系统形态探析

报告作者签名：王若冰

指导教师签名：赵新奎

完成日期 2025.11.02

目录

1 引言

随着计算规模的爆炸式增长、硬件形态的深刻变革以及应用需求的日益多样化，以 Linux 为代表的传统单体操作系统正在资源效率、扩展性、安全隔离和管理复杂性等方面面临着前所未有的挑战[1]。异构硬件的广泛应用和硬件资源解耦的兴起使得传统服务器的单体模型在资源利用率、硬件弹性和故障隔离方面捉襟见肘；大规模并行计算和无服务器计算的兴起对操作系统的状态管理、任务调度和数据溯源提出了更高的要求；此外，传统操作系统固化的安全与隔离策略在面对应用程序多样化的安全和性能权衡需求时过于刚性，无法对层出不穷的新型硬件隔离机制进行灵活应对。

面对上述挑战，学术界和工业界正在探索操作系统的全新形态，而 LegoOS[2]、DBOS[3] 和 FlexOS[4] 便是其中代表性的实践。LegoOS 提出的 Splitkernel 架构面向硬件资源解耦的未来，将操作系统功能分解到松耦合的硬件监视器上，以期实现极致的资源弹性和故障隔离；DBOS 倡导以数据库为中心重塑操作系统，将所有系统状态都视作数据库中的表，并利用分布式事务数据库来应对集群规模的复杂管理，同时提供内置的可观测性和高可用性；FlexOS 侧重于灵活隔离，通过构建模块化的 LibOS，允许在编译或部署时为应用定制安全与隔离策略，从而在多样化的安全需求和性能开销之间寻找最佳平衡点。

上述前沿探索共同揭示了未来操作系统的核心趋势，即告别一刀切式的单体架构，走向高度专业化、模块化、数据化和可定制化。通过硬件协同设计和灵活的安全边界以适应资源解耦的物理现实并满足不同应用的特定需求。本报告将基于对 LegoOS、DBOS 和 FlexOS 的深入剖析，探讨和展望操作系统未来的整体发展方向，并分析在大模型研究领域中，操作系统可能演进的新形态，整体组织架构如图1.1所示。

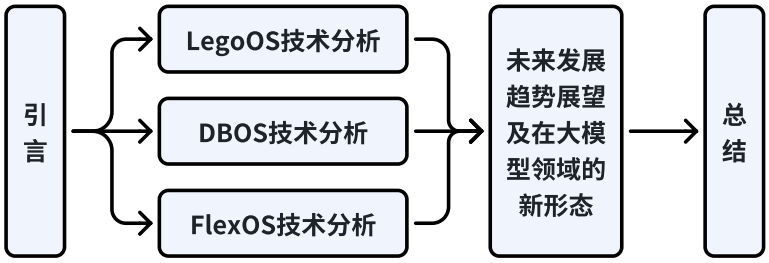

图 1.1 文章组织架构图

2 LegoOS：基于硬件资源解耦的分裂内核架构

在传统的单体服务器模型中，服务器是部署、操作和故障的基本单位，CPU、内存、存储等资源被紧密捆绑在同一主板上。在面对现代数据中心应用时，该范式存在四个核心问题：一是资源利用率低下，由于作业的 CPU 和内存需求必须在同一台物理机上得到满足，导致了严重的箱柜打包难题和巨大的资源浪费；二是硬件弹性差，在服务器封装完毕后几乎不可能按需对 CPU、内存等组件进行独立增减；三是故障域过大，任何一个关键组件故障都可能导致整台服务器宕机并影响其上运行的所有应用；四是异构性支持不良，将 GPU、FPGA 等新型硬件集成到现有的服务器主板代价过高，从而严重阻碍了新硬件的快速部署。为解决这些问题，硬件资源解耦应运而生，其核心思想是打破单体服务器的物理束缚，将 CPU、内存、存储等转变为通过高速网络互连的独立、可按需组合的硬件组件，而 LegoOS 为实现管理一个所有资源都分布在网络上的“分裂式”系统提供了可能，其核心组件架构如图2.1所示。

LegoOS 的核心贡献是提出了 Splitkernel 这一全新的操作系统架构，以“硬件被解耦时，操作系统也应当随之解耦”作为核心理念，Splitkernel 将传统操作系统中进程管理、内存管理、存储管理等核心功能分解为一系列松耦合的监视器，每个监视器运行在通过网络消息传递进行通信的独立硬件组件上。LegoOS 显式放弃了跨组件的缓存一致性, 因为在数据中心规模上维护一致性的开销是无法承受的。

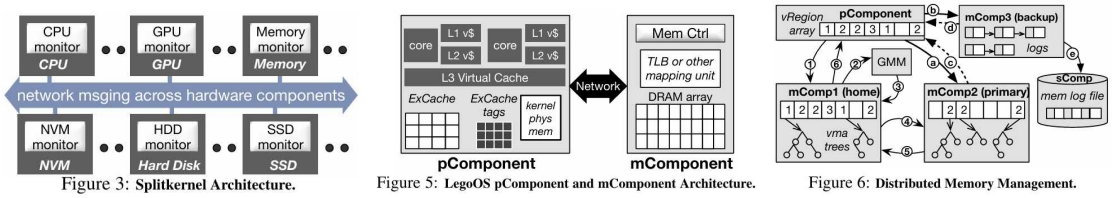

Figure 3: **Splitkernel Architecture.** Figure 5: **LegoOS pComponent and mComponent Architecture.** Figure 6: **Distributed Memory Management.**

图 2.1 LegoOS 核心组件架构图

为面对处理器与内存分离这一最大的挑战，LegoOS 提出了一种创新的软硬件协同设计方案。具体而言，LegoOS 首先将包括页表、TLB 在内的所有内存管理硬件功能全部移至内存组件；面对处理器组件无法访问物理内存地址的问题，其本地的各级缓存均被设计为虚拟缓存，并利用虚拟地址进行索引和标记；为了弥合网络访问（微秒量级）

和本地 DRAM 访问（纳秒量级）之间巨大的性能鸿沟，LegoOS 在处理器组件本地保留了少量 DRAM，这部分空间不作为主内存使用，而是被组织为一个巨大的、操作系统软件管理的扩展缓存 ExCache。当发生 ExCache 命中时，硬件对其进行处理，从而获得接近本地内存访问的性能，当 ExCache 未命中时，将触发一次 Trap 并由进程监视器介入以通过网络向相应的内存组件发起请求，获取数据块并填充回 ExCache；此外，LegoOS 的内存监视器实现了一套两级分布式虚拟内存管理机制，以虚拟区域为单位进行粗粒度的负载均衡，并通过将存储缓存放置在内存组件而非存储组件上，以获得更高的性能表现。

LegoOS 的评估结果证明了 Splitkernel 架构的可行性，在工作集大小适合的情况下，其性能仅比拥有足够本地内存的单体 Linux 务器慢 1.3 倍至 1.7 倍，而与那些内存受限并依赖本地 SSD 或网络交换的 Linux 相比，其性能要快 0.8 倍至 3.2 倍，显示了其对远程内存的主动管理远优于被动的页交换；此外，LegoOS 显著改善了资源打包效率，并由于其更小的故障域将系统的平均无故障时间提升了 17%-49%。LegoOS 的成功从操作系统层面完整地证明了硬件资源解耦这一未来数据中心架构的可行性，并在微秒级延迟的牺牲下换取数据中心在资源池化、弹性和故障隔离方面的巨大收益。其对 Linux ABI 的后向兼容使得现有应用无需修改即可运行，极大降低了未来架构迁移的门槛。

3 DBOS：以数据库为中心的操作系统

在面向复杂大规模分布式系统的状态管理任务时，源自 Unix“一切皆文件”的设计哲学已经完全无法适应现代数据中心的计算挑战。数据中心拥有的数万核心和 PB 级别的存储将操作系统的管理任务演变成大数据问题；无服务器计算的兴起和 GPU/TPU 等异构硬件的普及亦对调度和资源管理提出了新要求。当前数据中心的操作系统事实上是一个功能羸弱的单节点内核与一个复杂臃肿的集群管理所组成的集成度极差的两层结构，在这个脱节的架构中，要实现跨越调度以及文件、网络等子系统的全链路追踪几乎是不可能的，从而导致极端的调试、安全审计和合规性检查困难。

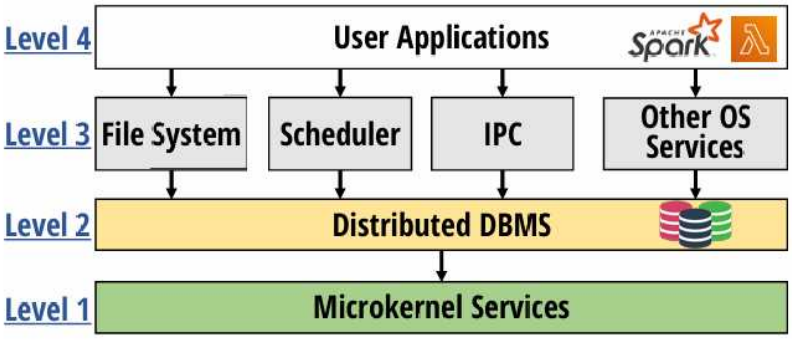

图 3.1 DBOS 系统架构图

为应对上述挑战，DBOS 提出了颠覆性的设想，相比在操作系统之上运行数据库的传统模式，DBOS 将操作系统构建在数据库之上，并用“一切皆表”的设计理念代替传统的“一切皆文件”的思想，其系统架构如图3.1所示。基于一个极简的微内核，DBOS 将一个高性能、分布式的事务型数据库置于操作系统的核心并使其成为事实上的内核，所有传统的操作系统服务都被重新实现为运行在该数据库之上的 SQL 查询和事务程序。在任务调度方面，系统通过维护 Task 表和 Worker 表将其由传统的复杂状态机转换变成高效的数据库事务；在文件系统方面，系统目录结构、文件元数据甚至小文件数据本身都存储在数据库表中，使得创建或删除等元数据操作变为时间复杂度 O(1) 的数据库事务；在进程间通信方面，一个进程通过向信息数据表中插入记录来向另一个进程发送信息，接受者通过选择、删除等数据库操作进行信息的获取和进一步处理。通过上述策略，

DBOS 将状态管理这一操作系统最核心的职责完全委托给分布式数据库，并获得了优异的性能表现。

DBOS 的设计带来了三大革命性优势。其一是内建的可靠性与一致性，传统操作系统中调度器、文件系统等各个子系统都需要各自实现高可用和故障恢复方案，而在 DBOS 中事务、高可用、崩溃恢复等能力仅由底层的 DBMS 实现一次，而所有上层系统服务对这些特性进行了自动继承；其二是惊人的性能与简洁性，原型测试表明 DBOS 的性能相较传统操作系统极具竞争力，其调度器每秒可处理 75 万个任务，远超现有系统。其文件元数据操作性能达到 ext4 的 10 倍。此外，实现一个复杂的调度策略或文件系统分析工具的代码量从数百行 C++ 缩减到几行 SQL，极大降低了系统复杂度；其三是终极的可观测性，集群的所有状态都存储在数据库表中的特性极大方便了对系统的监控、调试和数据溯源，通过 SQL 查询即可获得传统方法极难获得的统计信息。DBOS 的成功实践表明当前操作系统的核心挑战已从硬件抽象转向分布式状态管理，其基于逻辑状态聚合的核心理念为未来的集群操作系统提供了新的启迪。

4 FlexOS：实现按需定制的灵活隔离

传统的单体内核或微内核操作系统在设计时就将其安全与隔离策略固定在了架构中，其代表性的实践为 Linux 采用的用户态和内核态两级特权分离和微内核操作系统采用的多地址空间隔离。然而面向应用需求日益多样化的现代计算环境中，上述刚性安全策略操作系统模型整面临着严峻的挑战。实践经验表明，刚性模型无法满足不同应用甚至一个应用的不同组件对安全和性能截然不同的权衡需求；此外，传统操作系统架构日益僵化的现状导致其无法快速重构以有效利用 CPU 制造商不断推出的硬件隔离原语；随着计算机技术的发展，以 Meltdown 为代表的硬件漏洞已经充分暴露了现有隔离机制的脆弱性[5]，但在单体内核中部署 KPTI 等缓解措施需要付出巨大的工程努力和性能代价。上述问题表明对能够打破刚性安全模型、在编译或部署时有能力为应用提供灵活和按需定制的隔离策略的操作系统的迫切需求。

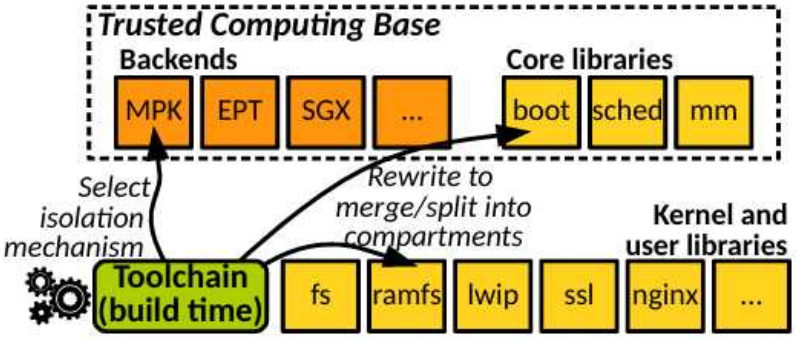

图 4.1 FlexOS 系统架构图

FlexOS 的核心思想是操作系统的隔离策略不应固定不变，而应面向应用需求变得更加灵活专精，通过一个基于 Unikraft 的模块化 LibOS 架构，结构如图4.1所示的 FlexOS 为上述问题提供了一个可行的解决方案。其的设计主要依赖于几个关键机制：一是细粒度的组件化，将调度器、TCP/IP 栈、文件系统等操作系统功能分解为细粒度的微型库；二是灵活的隔离边界，允许用户在构建时通过配置文件自由决定哪些组件被划分到哪个隔离舱中；三是可插拔的隔离原语，通过抽象出的一套通用 API，用户可在多种后端上定义隔离的边界并选择实现隔离的具体机制；四是硬件机制的底层支持，FlexOS 可以

使用 Intel MPK 或 EPT/VMs 实现轻量级或重量级的 intra-address-space 隔离；五是软件加固，FlexOS 支持为特定组件选择性开启以控制流完整性或地址消毒器为代表的软件加固功能。除去上述创新外，FlexOS 最核心的贡献是在源代码中使用抽象的“门”和“注解”来标记跨组件调用和共享数据，在编译时其工具链会根据用户的配置文件，将抽象注解特化为所选隔离机制的具体实现，从而实现安全策略与业务逻辑的解耦，有效提升开发效率与代码质量。

FlexOS 的评估结果证明了其设计的有效性，在 Redis 上的评估展示了其有能力生成数十种具有不同安全或性能特征的配置，充分证明了其灵活性的巨大价值；当 FlexOS 被配置为单体模式时，其性能与原生的 Unikraft 单体内核几乎完全相同，从而有效证明了其灵活性本身没有带来额外开销；更令人注意的是，为缓解配置组合爆炸式增长带来的调优困难，FlexOS 引入的部分安全排序技术能够帮助用户在给定的性能预算下自动化、概率性地筛选出最安全的一组配置。FlexOS 的成功实践有效证明了操作系统的安全隔离模型能够完成从固定的设计时决策向灵活的部署时选项转变，使得应用开发者可以根据其实际需求，为应用程序乃至其每个组件量身定制最合适的安全策略，而无需受限于操作系统设计者的僵化决策。

5 操作系统未来发展趋势及在大模型时代的新形态

LegoOS、DBOS 和 FlexOS 的成功不仅揭示了当前操作系统的设计困境，更从物理形态、逻辑形态和安全形态上清晰地指明了未来操作系统演进的三条关键技术路线。LegoOS 面向硬件资源解耦的物理趋势，通过分裂内核和 ExCache 的软硬件协同设计，表明未来的操作系统必须放弃对本地资源的僵化假设，从抽象单机硬件转变为一个高效的、数据中心规模的分布式硬件资源池管理者；DBOS 立足当前数据中心的核心痛点和极端规模下的分布式状态管理任务需求，通过将 OS 重塑为数据库使其获得了事务、高可用、崩溃恢复能力以及较强的可观测性，论证了 OS 成为数据中心级的状态管理器的可能；FlexOS 解决了传统刚性安全模型与应用多样化需求之间的根本矛盾，证明操作系统的安全和隔离模型不应再在设计时被僵化地固定，而应转变为可灵活定制的部署时选项。

综合上述启示，我们可以概括未来操作系统的三大发展趋势。一是专用化与模块化，单体通用内核已无法适应未来具有高度专用型的计算负载，为这些负载强行套用一个臃肿的通用 OS 是极其低效的，取而代之的将是基于 LibOS 或 Unikernel 思想的高度专用化 OS 实例；二是软硬件协同设计，操作系统不再仅仅是硬件的抽象层，而应主动释放硬件潜能，未来操作系统必须与 CXL 带来的内存池化、DPU/IPU 上的网络卸载、新型存储技术等硬件特性深度集成，通过共同设计以释放数据中心的潜力；三是“数据中心即计算机”的思想，操作系统的管理边界必须从单机扩展到整个数据中心，这意味着操作系统的核心职责从进程管理转向分布式状态管理和分布式资源编排。

当上述发展趋势与大规模人工智能相碰撞时，全新的操作系统形态正呼之欲出。近期人工智能特别是大模型的迅猛发展对操作系统提出了四大核心挑战：一是异构资源与拓扑盲目性，人工智能训练严重依赖 GPU/TPU 等加速器，传统 OS 对 GPU 资源与节点间拓扑关系认知不足，由此导致的拓扑盲目调度带来了昂贵的跨域数据传输并严重扼杀了潜在性能；二是海量数据的 I/O 瓶颈，PB 级别的训练数据吞吐使得传统 OS 中基于页缓存和内核协议栈的 I/O 路径延迟过高、开销过大；三是 GPU 显存的低效管理，传统操作系统对显存的管理能力严重不足，导致显存碎片化严重、跨进程共享困难，PyTorch 等应用层不得不自行实现复杂的显存管理，造成了极大的浪费和重复劳动；四是计算图

与进程错配，现有人工智能开发框架以计算图为单位执行，而操作系统只能理解进程或线程概念，无法感知流水线并行、张量并行等计算图的依赖关系，自然也无法进行有效的细粒度调度和资源分配。

面向上述需求，一个融合了前述三大趋势的新型操作系统可以从根本上为人工智能发展进行赋能。借鉴 LegoOS 的思想，可构建 GPU、HBM 等硬件资源池，允许一个训练任务按需、动态地拼装所需资源，从而极大提高异构硬件利用率；借鉴 DBOS 的思想，可将数据集、模型版本、GPU 状态等整个训练集群的状态全部置于数据库中管理，从而便于工程师像查询数据库一样进行数据溯源，定位导致问题的有毒数据并极大提升调试效率；借鉴 FlexOS 的思想，在多租户 MaaS 推理场景下，用户的模型可以运行在强隔离环境中以保证数据和模型安全，而在内部的高性能训练中切换到无隔离模式以获得极致的吞吐量。

在大模型成为核心计算负载的未来，操作系统的形态将发生根本性变革。为应对大模型生命周期各阶段的特异性需求，新型操作系统将聚焦调度原语变革，将张量或计算图节点作为基本的调度单位，并深度感知 GPU 拓扑和 HBM 容量，通过通信原语内核化和内存管理分层，为 AI 预训练提供硬件级别的优化支持和多层级内存显式管理。而大模型在性能和边缘设备部署能力方面的进展亦为 OS 提供了新的可能，通过在操作系统中内嵌一个小型 LLM，可实现意图驱动能力与自身状态的实时监控，从而依据用户意图和任务特点完成硬件资源全局分配的动态决策，并依据自身状态实时进行重构以动态调整隔离和性能策略。实践经验表明，大模型能够成为操作系统实现高级自省和自适应能力的引擎，并促进其完成从指令被动执行者向能够在巨大设计空间中进行寻优的主动智能调度中枢的转变。

6 总结

本报告剖析了传统单体操作系统在现代数据中心压力下正经历的深刻变革，并结合 LegoOS、DBOS 和 FlexOS 的成功实践展望了现代操作系统的发展趋势。LegoOS 通过分裂内核架构，展示了应对硬件资源解耦的可行路径，并实现了的资源弹性和故障隔离；DBOS 基于“操作系统即数据库”的颠覆性思想为大规模分布式状态管理提供了内建的可靠性与良好的可观测性；FlexOS 通过灵活隔离的做法将刚性安全策略转变为按需定制的部署时选项，实现了安全与性能的精细权衡。

上述工作清晰地指向了未来操作系统的共同趋势，即告别一刀切式的单体通用内核，走向专用化与模块化、软硬件协同设计以及数据中心级的状态管理。在面向大模型这一划时代的工作负载时，操作系统将进行调度核心转换并深度感知异构硬件拓扑，提供数据溯源并进行意图驱动的资源管理，同时不断增强自适应能力，甚至利用大模型实现自我演进，从而有效回应计算机科学界对下一代高效、智能、可信操作系统的需求。

参考文献

[1] ZHANG Y, ZHAO X, LI Z, et al. Integrating Artificial Intelligence into Operating Systems: A Comprehensive Survey on Techniques, Applications, and Future Directions[J]. ArXiv preprint arXiv:2407.14567, 2024.

[2] SHAN Y, HUANG Y, CHEN Y, et al. {LegoOS}: A disseminated, distributed {OS} for hardware resource disaggregation[C]//13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18). [S.l. : s.n.], 2018: 69-87.

[3] SKIADOPOULOS A, LI Q, KRAFT P, et al. DBOS: A dbms-oriented operating system[J]., 2021.

[4] LEFEUVRE H, BĂDOIU V A, JUNG A, et al. FlexOS: towards flexible OS isolation[C]//Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems. [S.l. : s.n.], 2022: 467-482.

[5] LIPP M, SCHWARZ M, GRUSS D, et al. Meltdown: Reading kernel memory from user space[J]. Communications of the ACM, 2020, 63(6): 46-56.