# Towards Practical Real-World Multikernel Operating Systems

Yauhen Klimiankou and Alexey Khoroshilov

klimenkov@ispras.ru

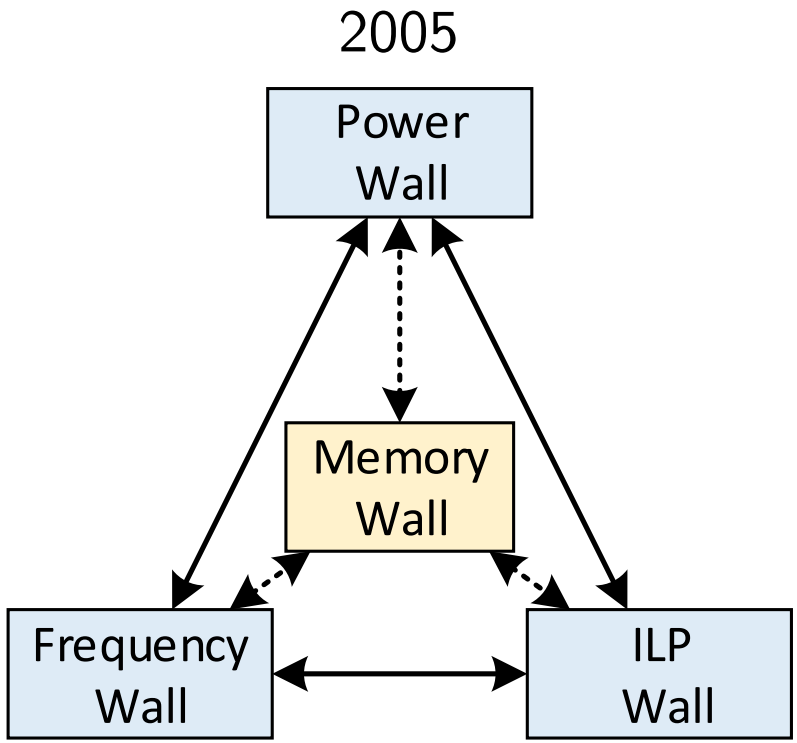
khoroshilov@ispras.ru

Ivannikov Institute for System Programming of the Russian Academy of Sciences

**OS DAY 2020**

5th November 2020

# Hardware Background: multi-core CPU




The architecture of the Multicore CPU became a reliable solution for the continuation of CPU performance growth.
During the last 15 years, multi-core became a dominating and regnant CPU design.

# Multikernel OS Architecture: Barrelfish

- **Barrelfish** is a Research OS built by ETH Zurich System Group & Microsoft Research and released in 2009.
- **Barrelfish** was designed and built to prototype and verify the **Multikernel OS** design.

Design principles:

1. Make all inter-core communication explicit.
2. Make OS structure hardware-neutral.
3. View the state as replicated instead of shared.

# Barrelfish

Computer with a multi-core CPU is a distributed system $\rightarrow$ OS is a network of interconnected OS nodes.

Motivation and Assumptions:

1. Systems are increasingly diverse.
2. Cores are increasingly diverse.
3. The interconnect matters.
4. Messages cost less than shared memory.
5. Cache coherence is not a panacea.
6. Messages are getting easier.
7. Traditional operating systems will not scale well on many-core computers.

Goals:

1. Scalability
2. Dealing with heterogeneous hardware

# Quest-V

The project at Boston University appeared in 2011 under Barellfish influence.



1. Goal is a high-confidence systems.
2. Quest-V targets real-time and safety-critical application domains.
3. Multikernel to provide extra reliability and fault tolerance.
4. Accompanies multikernel concepts by using hardware virtualization mechanisms.

Like Barrelfish, structure OS as a network of homogeneous special-purpose small kernels developed from scratch.

However, Quest-V does not support a concept of single image OS.

# Popcorn Linux


- Project started at Virginia Tech University in 2012.
- Aim is to apply the multikernel OS design to the traditional operating system based on the monolithic kernel, such as Linux.
- Replicated-kernel operating system.
- Software-based coherency based on inter-kernel communication asynchronous message passing.
- Replicated-kernels runs transparently applications developed for an SMP OS.
- Applications also can exploit replication at the user level.
- An alternative to virtual machines.
- A way to support cache-incoherent CPU and cores heterogeneity in Linux.

# Clustered Multikernel

- Project started in NICTA in 2012 to deal with scalability problems of verified seL4 microkernel.
- Multikernel for dealing with concurrency in the kernel via resource sharing avoidance.
- System statically divides into a set of clusters.
- By managing the number and size of the clusters, the appropriate trade-off between kernel scalability and application-level parallelism can be set, while kernel verification complexity stays preserved.
- Inter-cluster communication is explicit and bases on asynchronous message passing through the shared memory channel.
- Like in the case of Quest-V, clustered multikernel does not support the single image OS abstraction.



UNSW SYDNEY

# IHK/McKernel

- IHK/McKernel is a light-weight multikernel operating system created specifically for supercomputer environment.
- IHK/McKernel colocates in the single operating system side-by-side two kernels: IHK Linux and LWK McKernel.



Goals:

1. Provide scalable and consistent execution of large-scale parallel applications.
2. Rapidly adapt to exotic hardware and new programming models.
3. Provide efficient memory and device management.
4. Eliminate OS noise by isolating OS services in Linux and provide jitter-free execution environment on the McKernel.
5. Support the full POSIX API.

# Reevaluation of assumptions behind multikernel

- **Systems are increasingly diverse.** ✓
  - NetBurts and Itanium history support this viewpoint.
  - The hype of recent years around Spectre and Meltdown vulnerabilities shows the same point.
  - The growth rate of ISA features is concerning.
  - However, the multikernel OS design can provide only limited support in dealing with a problem of such kind. ✗
- **Cores are increasingly diverse.** ✗
  - The techniques for design, compilation, and deployment of applications on top of the heterogeneous computer system are still a big open challenge.
  - Focusing on fighting with core heterogeneity at the OS level is impractical for the current state of the market.

# Reevaluation of assumptions behind multikernel

- **The interconnect matters.** ✓
  - We already had a lesson learned from the distributed operating systems era history.
  - Attempts to stick to Single Image OS abstraction fall into the same trap.
  - Single Image OS abstraction is impractical for real-world application of multikernel operating systems and a wrong direction of development in general. ✗
- **Messages cost less than shared memory.** ✓
  - Barrelfish has proved this claim in practice in a series of experiments. In particular, experiments show scalability issues for cache-coherent shared memory even on a relatively small number of cores.

# Reevaluation of assumptions behind multikernel

- **Cache coherence is not a panacea.** ✗
  - There were significant doubts about cache coherency ability to scale in 2010.
  - However, ten years after, we still see only cache-coherent systems on the market.
  - AMD has released its 64-core Ryzen CPU at the end of 2019. It is now possible to assemble a server equipped by 256 physical cores or 512 logical cores.
- **Messages are getting easier.** ✓
  - Message-passing and event-driven programming model found extensive use in different domains and even programming languages.
  - Both are especially common in the area of distributed systems programming
  - It is entirely unclear why both Barrelfish and Popcorn Linux stick self to the Single Image OS model, which contradicts the distributed nature of applications based on message passing.

# Reevaluation of assumptions behind multikernel

- **Traditional operating systems will not scale well on many-core computers.** 
  - The assumption behind Barrelfish was next: traditional OS based on a monolithic kernel could not scale well with a count of cores growing.
  - However, currently, we can see that computers employing dozens of CPU cores become rather common, while they continue to use traditional OS pretty successfully.

# Missed value of Multikernel OS design

**Horizontal virtualization and software heterogeneity.**

- On the same machine, we can colocate different scheduling and interrupt handling policies, kernels and subsystems, providing different APIs, optimized for a particular range of workloads.
- We can consider the operating system node in the multikernel OS as a virtual machine or specialized container. By exploiting that observation, we can design, build, and deploy highly specific applications.
- Horizontal or soft virtualization poses a new point in the existing rapidly developing design space and blurs the border between the structuring of a particular trusted application and classical virtualization.

# Missed value of Multikernel OS design

**Reliability and Fault tolerance.**

- Multikernel OS resembles a distributed system and can inherit the decentralization property from it. A decentralized multikernel operating system does not have a single critical component.
- Multikernel OS relies on a kernel layer organized as an interlinked network of kernel nodes and distributes this criticality between all of them.
- There is a tremendous amount of techniques for reliability and fault tolerance accumulated in the domain of distributed systems in general and distributed operating systems in particular.
- There is an opportunity for very flexible and light speed recovery after node failures and minimization of applications data loss, which are impossible in case of computer networks.

# Missed value of Multikernel OS design

**Full dynamism of software stack.**

- Kernel in traditional vertically structured operating systems is completely static. Its on-the-fly replacement during system functioning is challenging, if not a wholly impossible task.
- Temporal dynamism. Multikernel systems can gradually and seamlessly evolve with time going by adding new features to the OS kernels, fixing bugs and security vulnerabilities inside them, and without the requirement to stop system functioning.
- Spatial dynamism. Multikernel systems can rapidly adapt and optimize themselves to the changing workloads.
- OS becomes able to change itself not only qualitatively, but also quantitatively.

# Summary

Multikernel can bring to our commodity computer systems flexibility, reliability, and dynamism of the level previously possible only in microkernel-based systems, and even extend them far beyond the borders of microkernel design.

The multikernel OS design creates a new huge multidimensional design space for operating system designers and researchers. And the researcher's community just started to explore new ground.

There is a set of valuable benefits that multikernel design can bring now on the current hardware and solve currently existing tasks.

# Bridging gap between multikernel OS and real world

- **Heterogeneous OS for homogeneous hardware instead of homogeneous OS for heterogeneous hardware.**
  - The network of kernels should include different kinds of OS kernels, some of which should be application specific and provide benefits for the resolution of actual real-world problems.
- **Extension of traditional OS to multikernel.**
  - Multikernel OS can employ traditional widely used OS kernel such as Linux or Windows to offload to it all non-specific to application activities.
  - Multikernel can use such OS kernel as a big universal driver for all hardware.
  - Multikernel can employ a traditional OS kernel as a service provider and offload complex tasks to it.
  - The traditional OS kernel can serve as a system monitor and toolbox for the administration of the host computer.
  - Finally, the traditional OS kernel can play of role of multikernel OS bootstrapper.

# Bridging gap between multikernel OS and real world

- **Rejection of Single Image OS as a fundamental abstraction.**
  - Multikernel OS implementations take into service Single Image OS as a fundamental abstraction mainly to deal with a lack of software.
  - Heterogeneity of multikernel OS through the employment of traditional OS kernel solves the same issues with a lack of software, but more elegantly.
  - Uncritical third-party applications can run on their natural environment isolated on the dedicated traditional OS node, while the special-purpose kernels and applications implemented from scratch are both designed as distributed systems and form the rest of multikernel network.

# Bridging gap between multikernel OS and real world

- **Special-purpose computer systems as a primary target for applications of multikernel OS ideas.**
  - Widespread adoption of multikernel design in the general-purpose OS seems challenging and will require tremendous investments of efforts not only in the operating system itself but also in runtimes, libraries, programming languages, and models.
  - A wide range of current special-purpose computer systems already can employ multikernel with relatively small investments and gaining valuable benefits.
  - Development of multikernel design on the ground of special-purpose computers in a long term perspective can provide a smooth transition for the new architectures for both hardware and software vendors, stretching over time and between vendors the required investments.

# Questions?