Nowadays, single-chip cache-coherent many-core processors having up to 100 cores are a reality. Many-cores with hundreds or even a thousand of cores are planned in the near future. In theses architectures, the question of the locality of L1 cache-miss related traffic (data, instruction and TLB) is essential for both scalability and power consumption (energy by moved bit). Our thesis is that: (i) handling the locality of memory accesses should be done at kernel level of an operating system in a transparent manner to user applications; and (ii) the current monolithic kernels are not able to enforce the locality of memory accesses of multi-threaded applications, because the concept of thread in these kernels is inherently unsuitable for many-core processors. Therefore, we believe that the evolution approach of monolithic kernels undertaken until now is insufficient and it is imperative to put the question of the locality of memory accesses in the heart of this evolution.

To prove our thesis, we designed and implemented ALMOS (Advanced Locality Management Operating System), an experimental operating system based on a distributed monolithic kernel. ALMOS has a new concept of thread, called Hybrid Process. It allows its kernel to enforce the locality of memory accesses of each thread. The resources (cores and physical memory) management in ALMOS's kernel is distributed enforcing the locality of memory accesses when performing system services. Decision making regarding memory allocation, tasks placement and load balancing in ALMOS's kernel is decentralized, multi-criteria and without locking. It is based on a distributed infrastructure coordinating, in a scalable manner, the accesses to resources.

Using the cycle accurate and bit accurate virtual prototype of TSAR many-core processor, we experimentally demonstrated that: (i) performance (scalability and execution time) on 256 cores of the distributed scheduling scheme of ALMOS's kernel outperform those of the shared scheduling scheme found in existing monolithic kernels; (ii) distributed realization of the fork system call enables this system service to scale on 512 cores; (iii) updating the distrusted decision-making infrastructure of ALMOS's kernel costs just 0.05 % of the total computing power of TSAR processor; (iv) performance (scalability, execution time and remote traffic) of memory affinity strategy of ALMOS's kernel, called Auto-Next-Touch, outperform those of two existing strategies First-Touch and Interleave on 64 cores; (v) concept of Hybrid Process of ALMOS's kernel scales up two existing highly multi-threads applications on 256 cores and a third one on 1024 cores; and finally (vi) the couple ALMOS/TSAR (64 cores) gives systematically much better scalability than the couple Linux/AMD (Interlagos 64 cores) for 8 multi-threads applications belonging to HPC and image processing domains.