may be implemented as a microprocessor, microcontroller, application specific integrated circuit (ASIC), discrete logic, or a combination of other type of circuits or logic. Similarly, memories may be DRAM, SRAM, Flash or any other type of memory. Flags, data, databases, tables, entities, and other data structures may be separately stored and managed, may be incorporated into a single memory or database, may be distributed, or may be logically and physically organized in many different ways. The components may operate independently or be part of a same program. The components may be resident on separate hardware, such as separate removable circuit boards, or share common hardware, such as a same memory and processor for implementing instructions from the memory. Programs may be parts of a single program, separate programs, or distributed across several memories and processors.

The respective logic, software or instructions for implementing the processes, methods and/or techniques discussed throughout this disclosure may be provided on computer-readable media or memories or other tangible media, such as a cache, buffer, RAM, removable media, hard drive, other computer readable storage media, or any other tangible media or any combination thereof. The tangible media include various types of volatile and nonvolatile storage media. The functions, acts or tasks illustrated in the figures or described herein may be executed in response to one or more sets of logic or instructions stored in or on computer readable media. The functions, acts or tasks are independent of the particular type of instructions set, storage media, processor or processing strategy and may be performed by software, hardware, integrated circuits, firmware, micro code, or any type of other processor, operating alone or in combination. Likewise, processing strategies may include multiprocessing, multitasking, parallel processing and/or any other processing strategy known now or later discovered. In one embodiment, the instructions are stored on a removable media device for reading by local or remote systems. In other embodiments, the logic or instructions are stored in a remote location for transfer through a computer network or over telephone lines. In yet other embodiments, the logic or instructions are stored within a given computer, CPU, GPU, or system.

A second action may be said to be "in response to" a first action independent of whether the second action results directly or indirectly from the first action. The second action may occur at a substantially later time than the first action and still be in response to the first action. Similarly, the second action may be said to be in response to the first action even if intervening actions take place between the first action and the second action, and even if one or more of the intervening actions directly cause the second action to be performed. For example, a second action may be in response to a first action if the first action sets a flag and a third action later initiates the second action whenever the flag is set.

To clarify the use of and to hereby provide notice to the public, the phrases "at least one of <A>, <B>, . . . and <N>" or "at least one of <A>, <B>, <N>, or combinations thereof" or "<A>, <B>, . . . and/or <N>" are defined by the Applicant in the broadest sense, superseding any other implied definitions hereinbefore or hereinafter unless expressly asserted by the Applicant to the contrary, to mean one or more elements selected from the group comprising A, B, . . . and N. In other words, the phrases mean any combination of one or more of the elements A, B, . . . or N including any one element alone or the one element in combination with one or more of the other elements which may also include, in combination, additional elements not listed.

While various embodiments of the innovation have been described, it will be apparent to those of ordinary skill in the art that many more embodiments and implementations are possible within the scope of the innovation. Accordingly, the innovation is not to be restricted except in light of the attached claims and their equivalents.

What is claimed is:

1. A memory appliance comprising:

a communication interface configured to communicate over a network;

a processor; and

a memory configured in an address space addressable by the processor, wherein the processor is configured to:

memory map at least a portion of a file to a memory region included in the memory, wherein a virtual address addressable by the processer is generated, and the at least the portion of file is accessible through the memory region at the virtual address; and

register the virtual address with the communication interface, wherein registration of the virtual address provides client-side memory access to the memory region, wherein the communication interface is configured to access to the memory region for a client of the memory appliance in a client-side memory access operation conforming to a memory access protocol,

wherein the memory region at the memory appliance is allocated as external memory at the client, wherein the external memory is memory treated as primary memory at the client, and wherein the client is configured to allocate, in response to a request at the client to allocate a portion of the external memory, a subset of the memory region to which the at least the portion of the file is mapped.

2. The memory appliance of claim **1**, wherein the memory appliance is a first memory appliance and the file is stored on a second memory appliance external to the first memory appliance.

3. The memory appliance of claim **1**, wherein the processor is further configured to move data in a portion of the memory region from a first memory tier to a second memory tier.

4. The memory appliance of claim **1**, wherein the processor is further configured to track and mark as dirty any portion of the memory region that is written to via an observable write, and write dirty portions of the memory region to the file, but portions of the memory region that are not dirty are not written to the file.

5. The memory appliance of claim **1**, wherein the processor is further configured to perform a batch portion invalidation and/or reclaim of a plurality of portions of the memory region and indicate to the communication interface that the portions of the memory region are reclaimed and/or invalidated.

6. The memory appliance of claim **1**, wherein a memory mapping of the at least the portion of the file includes an invocation of a memory-mapped interface.

7. A system comprising:

a client comprising a first communication interface, a first processor, and a first memory addressable by the first processor; and

a memory appliance comprising a second communication interface, a second processor, and a second memory addressable by the second processor, wherein a memory region at the memory appliance is included in the second memory, wherein the second communication interface is configured to access the memory

region on behalf of the client in a client-side memory access operation conforming to a memory access protocol,

wherein the memory region at the memory appliance is allocated as external memory at the client, wherein the external memory at the client is at the memory appliance but treated as primary memory at the client, and wherein the first processor is further configured to map the external memory to a first virtual address at the client and to allocate, at the client in response to a request to allocate a portion of the external memory, a subset of the memory region mapped to the first virtual address, and wherein data in the portion of the external memory is accessible by the client via the client-side memory access operation.

8. The system of claim **7**, wherein the second processor is configured to map the memory region to a second virtual address addressable by the second processor, and to register the second virtual address with the second communication interface, wherein registration of the second virtual address provides client-side memory access to the memory region.

9. The system of claim **8**, wherein the second processor is configured to allocate one or more portions of the memory region after registration of the second virtual address with the second communication interface.

10. The system of claim **8**, wherein the second processor is configured to avoid reclaiming one or more portions of the memory region in response to a request to pin the one or more portions.

11. The system of claim **8**, wherein the second processor is configured to memory map at least a portion of a file to the memory region, wherein the at least the portion of file is accessible by the second processor through the memory region at the second virtual address.

12. The system of claim **11**, wherein the file includes a first file, the memory region includes a first memory region, the second memory of the memory appliance includes a second memory region, and the second processor is further configured to:

memory map at least a portion of a second file to the second memory region and the at least the portion of the second file is accessible through the second memory region at a third virtual address by the second processor;

create a copy-on-write or copy-on-access mapping between the first file and the second file; and

register the third virtual address with the second communication interface, wherein registration of the third virtual address provides client-side memory access to the second memory region, wherein the second processor is further configured to initialize an uninitialized portion of the first memory region by copying data from the second file and/or from the second memory region in response to a client-side memory access request.

13. The system of claim **12**, wherein the client-side memory access request is a request to write to the first memory region.

14. The system of claim **12**, wherein the client-side memory access request is a request to access the first memory region.

15. The system of claim **12**, wherein the second virtual address is the same as the third virtual address, and wherein the copy-on-write or copy-on-access mapping is created in response to a fork of a child process from a parent process.

16. The system of claim **15**, wherein the first file is the same as the second file; the at least the portion of the second file is mapped to the third virtual address for the child process based on the child process being forked from the parent process; and the second processor is further configured to write data from the first memory region and the second memory region to the first file or the second file.

17. A method comprising:

memory mapping at least a portion of a file to a memory region of a memory appliance by invoking a memory-mapped interface, wherein the at least the portion of file is accessible through the memory region at a virtual address by a processor of the memory appliance;

enabling client-side memory access to the memory region by registering the virtual address with a communication interface of the memory appliance, wherein the communication interface is configured to access to the memory region on behalf of a client of the memory appliance in a client-side memory access operation conforming to a memory access protocol; and

allocating, at the client in response to a request at the client to allocate a portion of the external memory, a subset of the memory region to which the at least the portion of the file is mapped, wherein at least the subset of the memory region is treated as primary memory at the client.

18. The method of claim **17** further comprising bypassing the processor and accessing an initialized portion of the memory region with the communication interface in response to a client-side memory access request that is for the initialized portion of the memory region, and initializing an uninitialized portion of the memory region in response to a client-side memory access request that is for the uninitialized portion of the memory region.

19. The method of claim **18**, wherein the initializing of the uninitialized portion includes setting the uninitialized portion to one value.

20. The method of claim **18**, wherein the initializing of the uninitialized portion includes copying of data from the file to the uninitialized portion.

21. The method of claim **18**, wherein the initializing of the uninitialized portion includes copying of data from a second memory appliance and/or from a backing store to the uninitialized portion.

22. The method of claim **18**, wherein the initializing of the uninitialized portion is in response to an I/O fault or a page fault for the client-side memory access.

23. The method of claim **18**, wherein the initializing of the initialized portion is in response to a request to prefetch and/or pin the initialized portion.

24. The method of claim **18**, wherein treating, by the communication interface, a plurality of portions of the memory region as uninitialized based on a setting and/or a clearing of one or more indicators indicative of presence and/or access permission of the portions of the memory region.

* * * * *