Barrier-Enabled IO Stack for Flash Storage

A lot of storage venders has been provided disk device with a software for scheduling I/O requests issued by application. It prevents that developer do not control the IO scheduling at the device layer. So the ordering what application expect is not matched to the actual ordering proceeded in storage layer. There are three reason of it. The first can be generated when IO scheduler enqueue the application's IO request to the dispatch queue. The Second circumstance is that the requests are put into the command queue from dispatch queue. The last is occurred when requests cached to the storage owned cache from command queue. Today applications guarantees ordering of IO request by using fsync() system call. Because it block the other IO requests inserted to IO scheduling until previous IO requested processed, fsync() will decrease performance and concurrency. To resolve this problem, authors provided Barrier-enabled IO stack which is different from modern IO stack. They implemented some system call to help preserving ordering of IO requests. Using those system call, authors design BarrierFS, Order-preserving Block Device Layer, and Barrier-enabled Storage.