WHAT IS CLAIMED IS:

1. A computer-implemented method, comprising:

receiving, with at least one processor, multi-threaded log data comprising a plurality of logs, a plurality of markers, and a plurality of thread identifiers, each respective log of the plurality of logs associated with a respective marker of the plurality of markers and a respective thread identifier of the plurality of thread identifiers, each respective marker indicating that the respective log is in a respective category of a plurality of categories comprising a first category, a second category, and a third category, each respective thread identifier indicating that the respective log is associated with a respective thread of a plurality of threads;

for each respective log of the plurality of logs:

setting, with at least one processor, the respective thread identifier of the respective log as a most recently used item in a thread reference cache;

determining, with at least one processor, a respective log cache of a plurality of log caches in a map data structure based on the respective thread identifier for the respective log, wherein a respective key for the map data structure is based on the respective thread identifier, and wherein the respective log cache is a respective value associated with the respective key;

adding, with at least one processor, the respective log to the respective log cache in the map data structure;

determining, with at least one processor, whether to communicate at least one of the respective log or the respective log cache to a first repository based on the respective marker for the respective log, wherein determining whether to communicate the at least one of the respective log or the respective log cache comprises:

determining to communicate the respective log cache to the first repository when the respective marker indicates that the respective log is in the first category;

determining to communicate the respective log to the first repository when the respective marker indicates that the respective log is in the second category; or

determining not to communicate the respective log or the respective log cache to the first repository when the respective log is in the third category;

communicating, with at least one processor, the respective log to a second repository, wherein the first repository comprises a higher-speed repository and the second repository comprises a lower-speed repository;

determining, with at least one processor, whether to remove an oldest log from the respective log cache based on at least one of a log cache size limit or a time limit; and

determining, with at least one processor, whether to remove a least recently used log cache from the map data structure based on at least one map data structure size limit, wherein the least recently used log cache is associated with a least recently used item in the thread reference cache.

2. The computer-implemented method of claim 1, wherein the first repository comprises a cloud repository, and

wherein the second repository comprises an internal repository separate from the cloud repository.

3. The computer-implemented method of claim 1, wherein the first repository comprises a cache, and

wherein the second repository comprises a persistent storage database.

4. The computer-implemented method of claim 1, wherein the second repository comprises a remote cache.

5. The computer-implemented method of claim 1, wherein receiving the multi-threaded log data comprises receiving the multi-threaded log data from a plurality of data generators, each respective thread identifier associated with the respective thread from a respective data generator of the plurality of data generators.

6. The computer-implemented method of claim 5, wherein, for each respective log of the plurality of logs, the respective data generator of the plurality of data generators sets the respective marker based on the respective category of the respective log.

7. The computer-implemented method of claim 1, wherein determining whether to remove the oldest log from the respective log cache based on the at least one of the log cache size limit or the time limit comprises:

determining that the oldest log from the respective log cache is older than the time limit;

determining that a number of logs of the respective log cache is greater than the log cache size limit; and

determining to remove the oldest log based on the oldest log being older than the time limit and the number of logs being greater than the log cache size limit.

8. The computer-implemented method of claim 1, wherein the at least one map data structure size limit comprises a lower map data structure size limit and a higher map data structure size limit, and wherein determining whether to remove the least recently used log cache from the map data structure based on the at least one map data structure size limit comprises:

determining that a number of log caches of the map data structure is greater than the higher map data structure size limit; and

determining to remove the least recently used log cache and each next least recently used log cache until the number of log caches of the map data structure is less than or equal to the lower map data structure size limit.

9. The computer-implemented method of claim 1, wherein a first array is initially named a buffer array and wherein a second array is initially named a transfer array, and wherein receiving the multi-threaded log data comprises:

receiving a first portion of the multi-threaded log data in the first array named the buffer array for a first buffer time period;

after the first buffer time period, renaming the first array the transfer array and renaming the second array the buffer array; and

receiving a second portion of the multi-threaded log data in the second array renamed the buffer array for a second buffer time period following the first buffer time period.

10. A system, comprising:

a first repository comprising at least one first processor;

a second repository comprising at least one second processor; and

a multi-threaded log system comprising a thread reference cache, a map data structure comprising a plurality of log caches, and at least one third processor, the at least one third processor configured to:

receive multi-threaded log data comprising a plurality of logs, a plurality of markers, and a plurality of thread identifiers, each respective log of the plurality of logs associated with a respective marker of the plurality of markers and a respective thread identifier of the plurality of thread identifiers, each respective marker indicating that the respective log is in a respective category of a plurality of categories comprising a first category, a second category, and a third category, each respective thread identifier indicating that the respective log is associated with a respective thread of a plurality of threads;

for each respective log of the plurality of logs:

set the respective thread identifier of the respective log as a most recently used item in the thread reference cache;

determine a respective log cache of the plurality of log caches in the map data structure based on the respective thread identifier for the respective log, wherein a respective key for the map data structure is based on the respective thread identifier, and wherein the respective log cache is a respective value associated with the respective key;

add the respective log to the respective log cache in the map data structure;

determine whether to communicate at least one of the respective log or the respective log cache to the first repository based on the respective marker for the respective log, wherein when determining whether to communicate the at least one of the respective log or the respective log cache, the at least one third processor is configured to:

determine to communicate the respective log cache to the first repository when the respective marker indicates that the respective log is in the first category;

determine to communicate the respective log to the first repository when the respective marker indicates that the respective log is in the second category; or

determine not to communicate the respective log or the respective log cache to the first repository when the respective log is in the third category;

communicate the respective log to the second repository, wherein the first repository comprises a higher-speed repository and the second repository comprises a lower-speed repository;

determine whether to remove an oldest log from the respective log cache based on at least one of a log cache size limit or a time limit; and

determine whether to remove a least recently used log cache from the map data structure based on at least one map data structure size limit, wherein the least recently used log cache is associated with a least recently used item in the thread reference cache.

11. The system of claim 10, wherein the first repository comprises a cloud repository, and

wherein the second repository comprises an internal repository separate from the cloud repository.

12. The system of claim 10, wherein the second repository comprises a remote cache.

13. The system of claim 10, further comprising a plurality of data generators;

wherein, when receiving the multi-threaded log data, the at least one third processor is configured to receive the multi-threaded log data from the plurality of data generators, each respective thread identifier associated with the respective thread from a respective data generator of the plurality of data generators; and

wherein, for each respective log of the plurality of logs, the respective data generator of the plurality of data generators is configured to set the respective marker based on the respective category of the respective log.

14. The system of claim 10, wherein, when determining whether to remove the oldest log from the respective log cache based on the at least one of the log cache size limit or the time limit, the at least one third processor is configured to:

determine that the oldest log from the respective log cache is older than the time limit;

determine that a number of logs of the respective log cache is greater than the log cache size limit; and

determine to remove the oldest log based on the oldest log being older than the time limit and the number of logs being greater than the log cache size limit,

wherein the at least one map data structure size limit comprises a lower map data structure size limit and a higher map data structure size limit, and wherein, when determining whether to remove the least recently used log cache from the map data structure based on the at least one map data structure size limit, the at least one third processor is configured to:

determine that a number of log caches of the map data structure is greater than the higher map data structure size limit; and

determine to remove the least recently used log cache and each next least recently used log cache until the number of log caches of the map data structure is less than or equal to the lower map data structure size limit.

15. The system of claim 10, further comprising:

a first array initially named a buffer array; and

a second array initially named a transfer array,

wherein, when receiving the multi-threaded log data, the at least one third processor is configured to:

receive a first portion of the multi-threaded log data in the first array named the buffer array for a first buffer time period;

after the first buffer time period, rename the first array the transfer array and rename the second array the buffer array; and

receive a second portion of the multi-threaded log data in the second array renamed the buffer array for a second buffer time period following the first buffer time period.

16. A computer program product comprising at least one non-transitory computer-readable medium including one or more instructions that, when executed by at least one processor, cause the at least one processor to:

receive multi-threaded log data comprising a plurality of logs, a plurality of markers, and a plurality of thread identifiers, each respective log of the plurality of logs associated with a respective marker of the plurality of markers and a respective thread identifier of the plurality of thread identifiers, each respective marker indicating that the respective log is in a respective category of a plurality of categories comprising a first category, a second category, and a third category, each respective thread identifier indicating that the respective log is associated with a respective thread of a plurality of threads;

for each respective log of the plurality of logs:

set the respective thread identifier of the respective log as a most recently used item in a thread reference cache;

determine a respective log cache of a plurality of log caches in a map data structure based on the respective thread identifier for the respective log, wherein a respective key for the map data structure is based on the respective thread identifier, and wherein the respective log cache is a respective value associated with the respective key;

add the respective log to the respective log cache in the map data structure;

determine whether to communicate at least one of the respective log or the respective log cache to a first repository based on the respective marker for the respective log, wherein when determining whether to communicate the at least one of the respective log or the respective log cache, the at least one third processor is configured to:

determine to communicate the respective log cache to the first repository when the respective marker indicates that the respective log is in the first category;

determine to communicate the respective log to the first repository when the respective marker indicates that the respective log is in the second category; or

determine not to communicate the respective log or the respective log cache to the first repository when the respective log is in the third category;

communicate the respective log to a second repository, wherein the first repository comprises a higher-speed repository and the second repository comprises a lower-speed repository;

determine whether to remove an oldest log from the respective log cache based on at least one of a log cache size limit or a time limit; and

determine whether to remove a least recently used log cache from the map data structure based on at least one map data structure size limit, wherein the least recently used log cache is associated with a least recently used item in the thread reference cache.