TimeBaseUUIDNotes

The time-based UUID generated by http://johannburkard.de/software/uuid/ implies that it is a conforming Type 1 UUID. It isn't exactly, which is probably a good thing. A Type 1 UUID consists of the following:

1. A timestamp consisting of a count of 100-nanosecond intervals since 00:00:00.00, 15 October 1582 (the date of Gregorian reform to the Christian calendar).
2. A version (which should have a value of 1).
3. A variant (which should have a value of 2).
4. A sequence number, which can be a counter or a pseudo-random number.
5. A "node" which will be the machines MAC address (which should make the UUID unique across machines).

The challenge with a UUID is to make it be unique for multiple processes running on a single machine and multiple threads running in a single process. The Type 1 UUID as specified above does neither. On a fast machine with multiple cores it is quite possible to have a UUID generated with the same time value. This can be remedied only if the sequence number can span threads and processes, something that is quite challenging to do efficiently.

The Time Based UUID referenced compensates for these issues by:

1. Only using the normal millisecond granularity returned by System.currentTimeMillis() and adjusting it to pretend to contain 100 ns counts. 2. Incrementing the time by 1 (in a non-threadsafe manner) whenever a duplicate time value is encountered. 3. Using a pseudo-random number associated with the UUID Class for the sequence number.

Incrementing the time by 1 allows multiple threads to uniquely create up to 10,000 UUIDs in the same millisecond in the same process. Using a pseudo-random number for the sequence number provides a 1 in a 16,384 chance that each UUID Class will have a unique id.

These mechanisms provide a reasonable probability that the generated UUIDs will be unique. However, the issues to be aware of are:

1. The computer is capable of generating more than 10,000 UUIDs per microsecond.
2. Applications creating UUIDs on different threads could get duplicates since the time is not incremented in a thread-safe manner.
3. More than one instance of the Class is in the VM in different Class Loaders - this will be mitigated by each Class having its own sequence number.
4. There is no guarantee that two instances of a UUID in the same or different VMs will have a different sequence number - just a reasonable probability that they will.