In GigaSpaces 16.4.3 and later, the use of initialLookupAttributes in the services.config file remains conceptually similar to earlier versions, but due to major internal refactoring (especially moving from legacy JavaSpaces/Jini APIs toward more modern and consistent configuration styles), some classes might be deprecated, changed in packaging, or behavior.

The key portion you're referring to is:

initialLookupAttributes =

new net.jini.core.entry.Entry[] {

new net.jini.lookup.entry.Name("Lookup"),

new org.jini.rio.entry.OperationalStringEntry("Service Grid Infrastructure")

};

**✅ Updated Approach for GigaSpaces 16.4.3:**

You can **retain this logic with minimal change** — GigaSpaces 16.4.3 still supports initialLookupAttributes using the same classes, **as long as the required JARs (like rio-lib, jsk-platform, etc.) are available in the classpath.**

**Suggested update:**

1. **Make sure** the following JARs are included in your classpath (these are typically found in $GS\_HOME/lib/platform or similar):
   * rio-lib-<version>.jar
   * jsk-platform.jar
   * jsk-lib.jar
2. Keep your block as-is in the services.config file (or if you are shifting to XML-based Spring-style bean config, it would need rewriting in bean form — but for services.config, this is fine):

initialLookupAttributes = new net.jini.core.entry.Entry[] {

new net.jini.lookup.entry.Name("Lookup"),

new org.jini.rio.entry.OperationalStringEntry("Service Grid Infrastructure")

};

1. Ensure your RIO\_HOME and GS\_HOME are properly configured so the classpath includes required Rio-related classes.

**⚠️ Notes on GigaSpaces 16.x Changes:**

* While much of the configuration is still backward-compatible, newer best practices encourage use of XML or programmatic configuration via Spring-style context files.
* If you're transitioning to **PU-based deployments or Spring Boot-based services**, these Jini-style configurations are typically moved into either the container startup or orchestrated via admin.config.