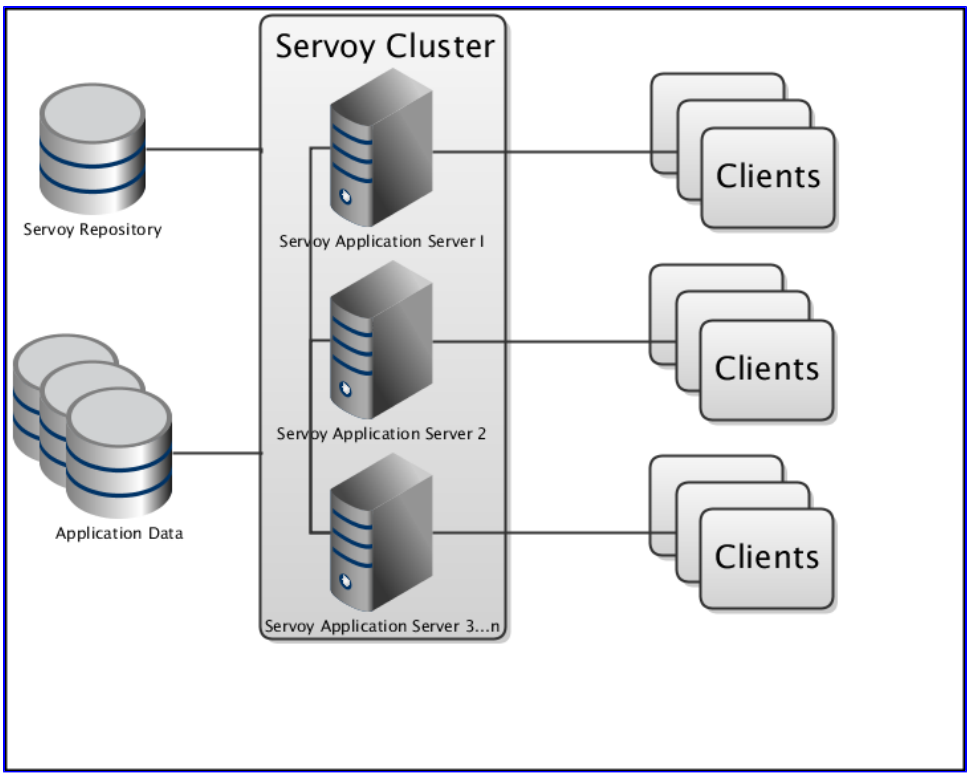
Clustering Servers

Terracotta deprecated DSO (Distributed Shared Objects)

Unfortunately Terracotta *deprecated* the technology that Servoy used to provide Servoy clustering (DSO). So clustering is only possible with older versions of Terracotta that still support DSO and only with Java 1.6 (not higher). That means that **Servoy 8** (compiled for Java 1.7) **will not work** at all **with Terracotta** clustering. There is a case in the support system for switching to other clustering tech. Servoy 7.x should still work with Java 1.6 and now deprecated Terracotta DSO.

Until this gets addressed: if your solution can be split into multiple completely independent servers (each user being served by the same Servoy Server all the time) you can try doing that - so keeping multiple separate independent Servoy Servers. Each server in this case has it's own independent subset of data. But this means you have to have subsets of users that don't need to interact with data stored on another server (so no data broadcasting/locking/... needs - no real clustering needs).

Servoy Cluster allows multiple Servoy Application Servers instances to be clustered together and act as 1 virtual Servoy Application Server.



Clustered Servoy Application Servers share the same Servoy Repository. This means that all Servoy Application Servers in the cluster share the same version of the solutions available. Servoy Cluster makes all Servoy Application Servers in the cluster remain in sync when importing (a new version of) a solution. Servoy's Databroadcasting mechanism also operates throughout the entire cluster.

Servoy Cluster is based on Terracotta , the world's most powerful and widely-adopted open source platform for Java scalability and availability.

This chapter describes the **set up of a cluster** in Cluster Installation..

A Servoy Cluster is best set up in combination with a Load Balancer. Adding a Load Balancer and a HTTP Server in front of the Servoy Cluster will provide the following benefits:

- Using an HTTP Server in front of the cluster provides Clients a **single point of entry**, hiding the individual Servoy Application Server instances from view for Clients.
- Using a Load Balancer in front of the cluster allows for the **automatic distribution of connecting clients** over the available Servoy Application Servers.

For a step by step guide on setting up Apache HTTP Server as single point of entry and Load Balancer, see Load Balancing Servers.