HTTP requests and static clusters

When incoming requests reach an HTTP server, the web server plug-in, which runs in-process with the HTTP server, decides how to handle these service requests. While some requests for static content can be serviced directly by the HTTP server, any requests for dynamic content, and some requests for static content, are sent to the back-end application servers. We refer to this process as *plug-in WLM*, as illustrated in Figure 15-11. For these WebSphere requests, high availability for the web container becomes an important piece of the failover solution.

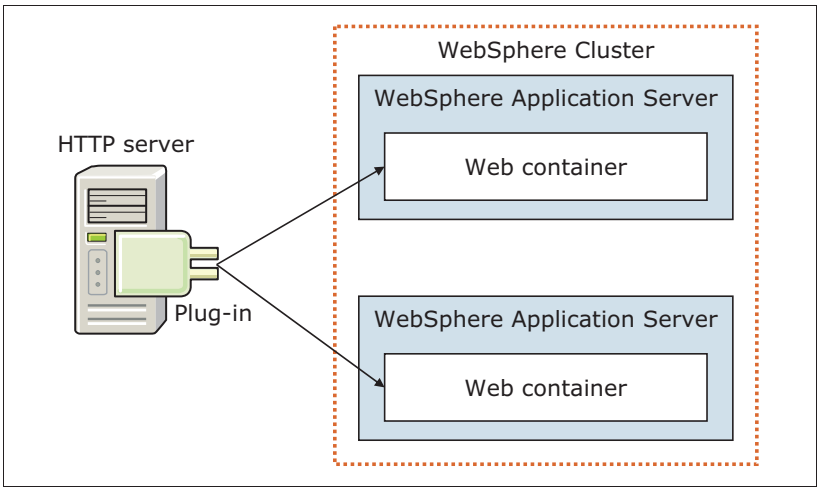

Figure 15-11 Plug-in workload management

WebSphere provides the following load balancing options:

- Round-robin

  This routing is based on the weight that is associated with the cluster members. If all cluster members have identical weights, the plug-in sends equal requests to all members of the cluster, assuming no strong affinity configurations. If the weights are scaled in the range 0 - 20, the plug-in routes requests to those cluster members with the higher weight value more often. No requests are sent to cluster members with a weight of zero (0) unless no other servers are available. Round-robin is the default load balance policy.

  Use the following formula as a guideline for determining routing preference:

  % routed to Server1 = weight1 / (weight1+weight2+...+weight*n*)

  Where there are *n* cluster members in the cluster.

- Random

  The plug-in picks a member of the cluster randomly.

The load balancing options are impacted by the session affinity. After a session is created at the first request, all the following requests have to be served by the same member of the cluster. The plug-in retrieves the application server that serves the previous request by analyzing the session identifier and tries to route to this server. We describe session management concepts in detail in Chapter 28, “Session management” on page 961.

HTTP requests and dynamic clusters

With the dynamic workload management features, the ODR becomes an important figure in workload management. It handles queuing and dispatching of incoming requests to the dynamic application server clusters, according to defined operational policies for optimum results and performance.