Data services are most prominent among back-end services. But there are other services too.

There is always a security service to log-in and authenticate users. DevForce establishes a security context during each initial client contact and attaches that context to all exchanges between the application server and the client. The client can use that context to authorize end user activity.

If the client is already logged in via ASP.NET Security, DevForce will repackage the ambient security context as its own. Your client application could perform its own ASP.NET login with Windows or Forms authentication. Or you can write a custom login process that does whatever you need it to do.

Applications often access external services such as Credit Scoring systems or Credit Card services. In general, the application clients shouldn't access such services directly as the communications typically involve sensitive data and license keys that must be secured on the server.

A common alternative and good practice is for the client to route a request for these services through the *EntityServer*. The channel between the client and the *EntityServer* has already been secured. You can leverage that fact and the server-side role information to screen requests from the client and filter the responses from the external service. The client places the request by invoking a custom remote service method on the server via the *EntityManager.InvokeServerMethod* as discussed here. Your server-side service method can do the authorization, filtering, and the real work of interacting with the external service or delegate any of those functions to helper components as circumstances require.

Your custom service methods can accept any number of parameters and can return any object. The only restriction is that parameter arguments and returned results be serializable. You can even send or return an arbitrary basket of entities in the form of an *EntityCacheState*.

You also may have proprietary logic that you don't want to deploy to client machines. You want to safe-guard such intellectual property in a secure server-side environment. Treat these as you would external services and use the same remote server method techniques to exercise them.