The OOTB impersonation:

1) Allows straightforward auditing of impersonation sessions via the ODBC audit logs (but not the text based audit log) by including both the impersonator's and the impersonatee's DNs in the ODBC audit log. No extra setup is required.
2) It disregards security levels of realms, instead it requires a rule in each realm allowing the impersonator to impersonate in the realm, and another rule allowing the impersonatee to be impersonated in each realm, and a policy tying the rules and users together. So it takes more configuration effort than the GSE Impersonation does.
3) It uses SMSAVEDSESSION cookie, so you have two large cookies being delivered to the web server with every transaction (although this could be modified to work like the GSE Impersonation Catalog Component).
4) It doesn't work yet with Federation.
5) You need to check with each ASA group concerning which Application Server Agents it will work with. The Weblogic group has sent out an email about how to get OOTB impersonation to work with Weblogic. And I believe the Websphere ASA is supposed to support it sometime in the summer of 2006.
6) OOTB definitely allows multiple attributes to be used for disambiguating the impersonatee's login ID.
7) The OOTB places the impersonator's DN in the User Context structure, so custom active expressions (rules, responses, and policies) can be developed to apply custom logic in addition to the OOTB features.
8) The OOTB places the impersonator's DN in the Event API AZ data structure so that custom event handlers can be written that can tell when an impersonation session is taking place. This could be used for a custom logger that logs only impersonation login and resource authorization events.

GSE Impersonation Catalog Component:

1) Allows auditing, but it is more complicated. You have to find the sessionID in the authentication transaction that starts the impersonation session, and then locate transactions via the session ID. The impersonation session authentication transaction includes both the impersonatorDN and the impersonatee DN.
2) If the impersonator hits a realm with a higher security level they will be rejected. If a company wants to be able to impersonate realms with high security levels, I recommend setting up two (or more) different impersonation realms with different security levels set for the impersonation auth scheme, and only authorize selected trusted impersonators access to the high security level impersonation realm.
3) Stores impersonator's original session in asp/JAVA session memory, so sticky bit needs to be set on front end switches so the impersonation sessions always

gets routed to the same web servers (or use a cluster that shares session memory somehow). Although my fcc's and asp's/jsp's could be modified to use SMSAVEDSESSION (@popsession) like the OOTB does.

4) Once an impersonation session has begun, SiteMinder can't distinguish it from a normal user session (but I do provide an optional cookie response and a header response capability to allow the Impersonator's loginID or DN to be delivered to apps so they can detect impersonation sessions). So it works with all the ASA agents and Federation (thanks to Aurangzeb for testing Federation).
5) The OOTB impersonation uses UserPolicies to determine who can be impersonated, while my solution module uses custom code that can determine who can be authorized based on comparing impersonator and impersonatee attribute values, DIT structure, and groups. I have found at least one case where my solution could be used to authorize users in a way that the OOTB couldn't (linked-account impersonation), but then again there may be ways that UserPolicies could authorize user's that I can't (eTelligent rules for instance). So that trade off would have to be looked at on a case by case basis.
6) Support for Delegated Impersonation is available, but as far as I know Delegated Impersonation cannot be done via the OOTB impersonation.
7) Works with all the Application Server Agents, whereas the OOTB impersonation doesn't work with the Websphere ASA the last time I checked.
8) Allows you to initiate a federated session, whereas you cannot initiate a federated session during an OOTB impersonation session.
9) Allows a user to have multiple, linked accounts (personnas) and manually switch their identity between the multiple, linked accounts without having to re-enter passwords. Automatic identity switching between default identities can also be configured.

Because of these differences, GSE is still making the GSE Impersonation Catalog Component available for a fee.