

DEVELOPING SCALABLE ENTERPRISE APPLICATIONS WITH SHAREPOINT SERVER 2007

Eric Shupps, Microsoft MVP [SharePoint]

Who Am I?

- Founder and President, BinaryWave
- Microsoft MVP (Windows SharePoint Services)
- ISPA Regional Evangelist (South Central US)
- SharePoint Development Patterns & Practices Advisory Board Member
- Blog: www.sharepointcowboy.com



Topics

- ◎ Data Access
 - Site Objects
 - Cross-Site Objects
 - External Objects
- ◎ Data Aggregation

Data Access

Site Objects

SharePoint Lists

- List data is stored in single UserInfo table for each content database
- List constructs are virtualized and non-relational (but you can create relationships programmatically)
- Lists are optimized for UI not for transactional data access
- Limited data types and data type instances

Data Retrieval

**Use an indexer
to return a
single list item
from the
collection**

Data Retrieval

~~Use an index
to return a
single list item
from the
collection~~

Data Retrieval

**Looping operations in
the object model are
very resource
intensive**

Data Retrieval

- Use CAML queries for targeted item selection
- Maintain SQL indexes for optimal query execution
- Manipulate item data using in-memory objects (SPListItemCollection, DataTable, <List>, XML)

Data Presentation

- 2000 item limit guideline applies only to built-in list views NOT to list contents
- Extremely large list structures (10,000+ items) require custom web parts to display content
- Data-bound objects provide best performance
- XML + XSLT outperforms list view objects
- Page results to reduce rendering times
- Beware caching of large data sets in ViewState

Web Services

- Provide front-end for OM code
- Return list objects as XML
- Manipulating XML in memory is more efficient than repeated calls OM
- Require extensive text manipulation
- Beware column name differences and content formatting

Web Services

SharePoint Emoticons

0;# SP sticking it's tongue out (:-P)

x0020 The_x0020_Final_x0020_Frontier

ows_ All Your Fields Are Belong to OWS

-1001 Look! Booleans can be anything!

DEMO

Using SharePoint Lists as Data Source

Data Access

Cross-Site Objects

CAML

- ⦿ No native cross-site functionality
- ⦿ Combine with recursive methods to create consolidated dataset
 - Recurse list of site and list objects (XML, Delimited, GetSubwebsForCurrentUser, etc.)
 - Store list items in ADO.NET object (DataTable, DataSet, etc.)

PortalSiteMapProvider

- MOSS only
- Optimized for repeated calls to same set of objects
- Use the *GetCachedListItemsByQuery()* method for cacheable items that do not change often
- Executes CAML Query without having to walk the collection of SPWeb objects
- List items must still be iterated using in-memory methods

Search

- ⦿ Provides non-contextual item data within defined scope(s)
- ⦿ Requires parsing potentially large number of results (XML or ResultTable)
- ⦿ Scopes and queries must be managed to return desired results – no guarantee that default results will include the desired item(s)
- ⦿ Access to contextual item data requires recursive object methods

Web Services

- ⦿ Some methods provide multiple object support (Lists, Webs, etc.)
- ⦿ Results are non-contextual; however, most metadata is available in native methods
- ⦿ Resultant XML must be parsed and manipulated
 - XML may be consolidated and XPATH used in place of recursion
- ⦿ Extensive/repetitive calls increase overhead on WFE's

DEMO

Retrieving SharePoint List Items from Multiple Site Collections using MOSS Search

Data Aggregation

Relational Data

- ◎ SharePoint lists are inherently non-relational
 - Structure in Lists table
 - Data in UserInfo table
 - Fixed number of field elements
- ◎ Relationships must be created programmatically using combination of list definitions and ADO.NET, LINQ, etc.

ADO.NET

- ⦿ Define list structures and primary/foreign keys
 - Remember data type limitations
- ⦿ Retrieve list objects via OM or Web Services
- ⦿ Create relational DataSet objects, merge into single DataTable
- ⦿ Use stand-beside DB for caching and native SQL queries

LINQ

- ⦿ Simplifies programming
- ⦿ Allows for normalized query language similar to T-SQL
- ⦿ Performance impact is dictated by type of operation
 - List Items = CAML Query
 - Site Objects = Recursive OM
 - XML = Iterative string parsing

Recursive Site Queries

- Avoid direct database queries
- Create Timer Job or Windows Service to execute recursion code
- Schedule for off-peak hours
- Store results in separate database
- Query against custom DB
- Batch bulk update tasks using web services

DEMO

Programming Relational List Objects

Best Practices

- ⦿ Site Objects – Use CAML to retrieve list objects
 - Create custom web parts to display large datasets
- ⦿ Cross-Site Objects – PortalSiteMapProvider and Search provide optimal performance
- ⦿ External Objects
 - SP to SP – Use web services
 - Other – Employ least overhead-intensive method
- ⦿ Data Aggregation – Relational list methods should be reserved for smaller datasets

Resources

- ◎ Slides and sample code:
 - www.sharepointcowboy.com, “Presentations” link
- ◎ Andrew Connell’s Building High Performance MOSS Solutions
 - <http://www.andrewconnell.com/blog/articles/Speaking.aspx#20080602a>
- ◎ SharePoint 2007 Performance and Scalability White Paper
 - <http://blogs.msdn.com/sharepoint/archive/2008/06/17/belated-announcement-sharepoint-server-2007-scalability-and-performance-whitepaper-now-available.aspx>
- ◎ TechNet IIS Performance Tuning
 - <http://www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/502ef631-3695-4616-b268-cbe7cf1351ce.msp?mfr=true>

THANK YOU FOR ATTENDING!

Please be sure to fill out your session evaluation!