

SHAREPOINT **SATURDAY**



Welcome

Life in the Fast Lane

SharePoint Performance Planning
and Optimization

Welcome to SharePoint Saturday Houston

Thank you for being a part of the 2nd Annual SharePoint Saturday Houston

- Please turn off all electronic devices or set them to vibrate.
- If you must take a phone call, please do so in the hall so as not to disturb others.
- Thanks to our Platinum Sponsors:



Information

- Speaker presentation slides will be available at SharePointSaturday.org/Houston within a week
- Keep checking website for future events
- The Houston SharePoint User Group at www.h-spug.org, will be having it's May meeting this Wednesday on May 11th. Please be sure to join us!
- Have a great day!

Who Am I?

Eric Shupps

- President, BinaryWave
- SharePoint Server MVP
- Member, Patterns & Practices Advisory Board (spg.codeplex.com)
- Web: www.binarywave.com
- Blog: www.sharepointcowboy.com
- Twitter: [@eshupps](https://twitter.com/eshupps)
- Facebook: www.facebook.com/sharepointcowboy



Agenda

- Infrastructure
- Databases
- Pages
- Lists
- Components



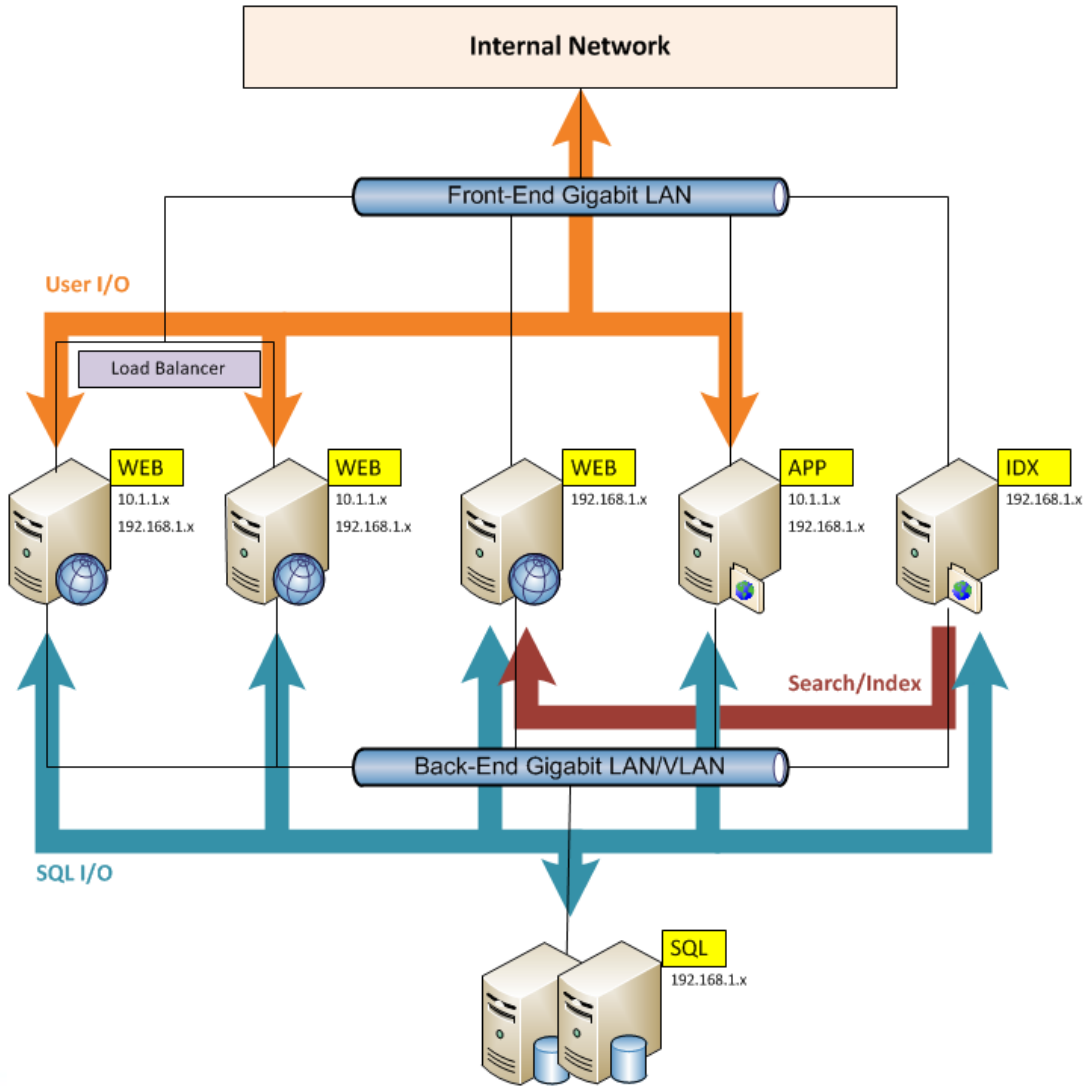
IT Professional



Developer

SERVERS

Network



Architecture



- Design to function
 - Heavy read operations require more web servers
 - Heavy write operations require increased SQL IOPS
 - Heavy services (i.e. Search) require additional application servers
- Design to Locality
 - Global distribution with heavy write may require localized farms



Web Servers



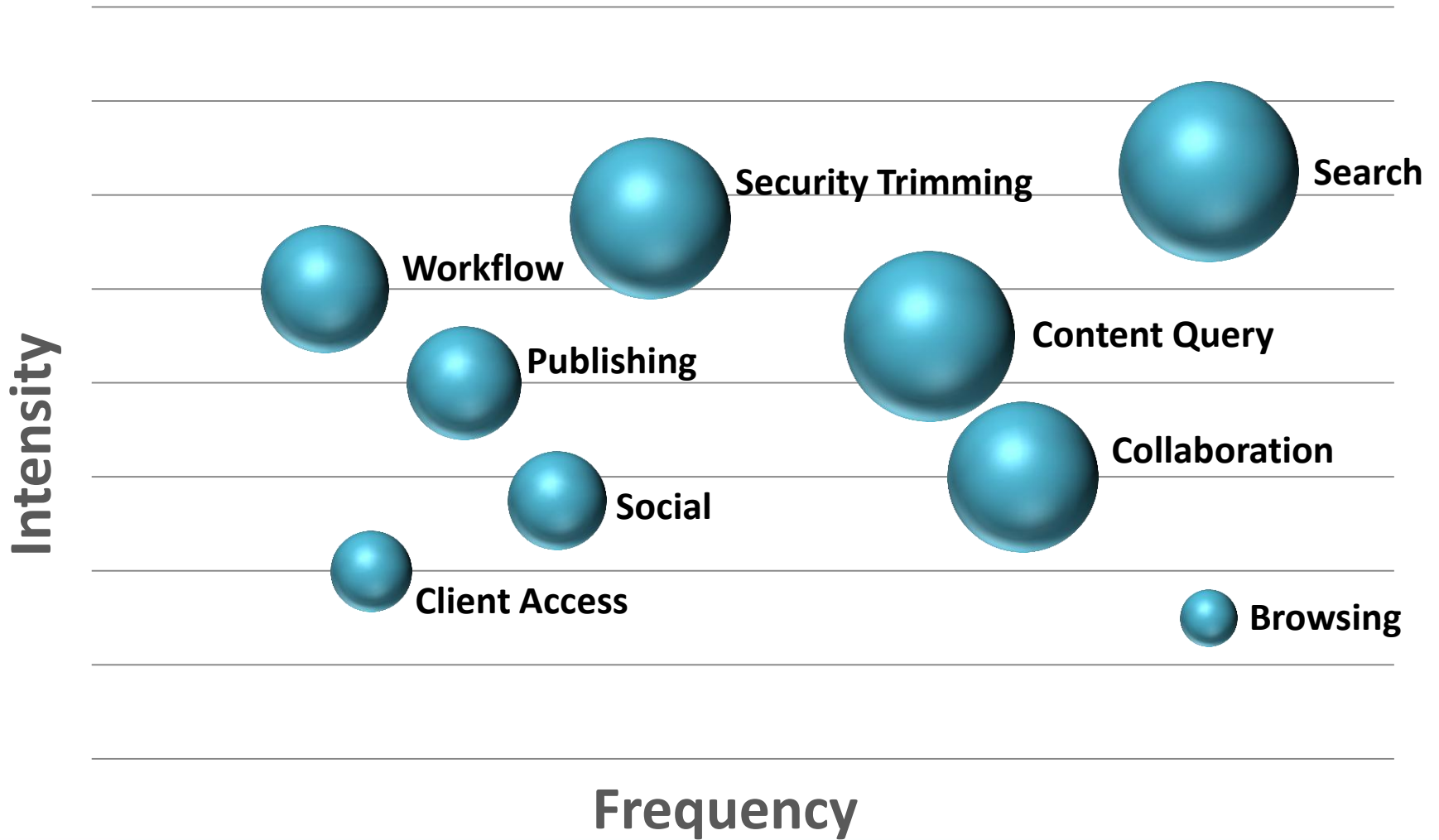
Application Servers



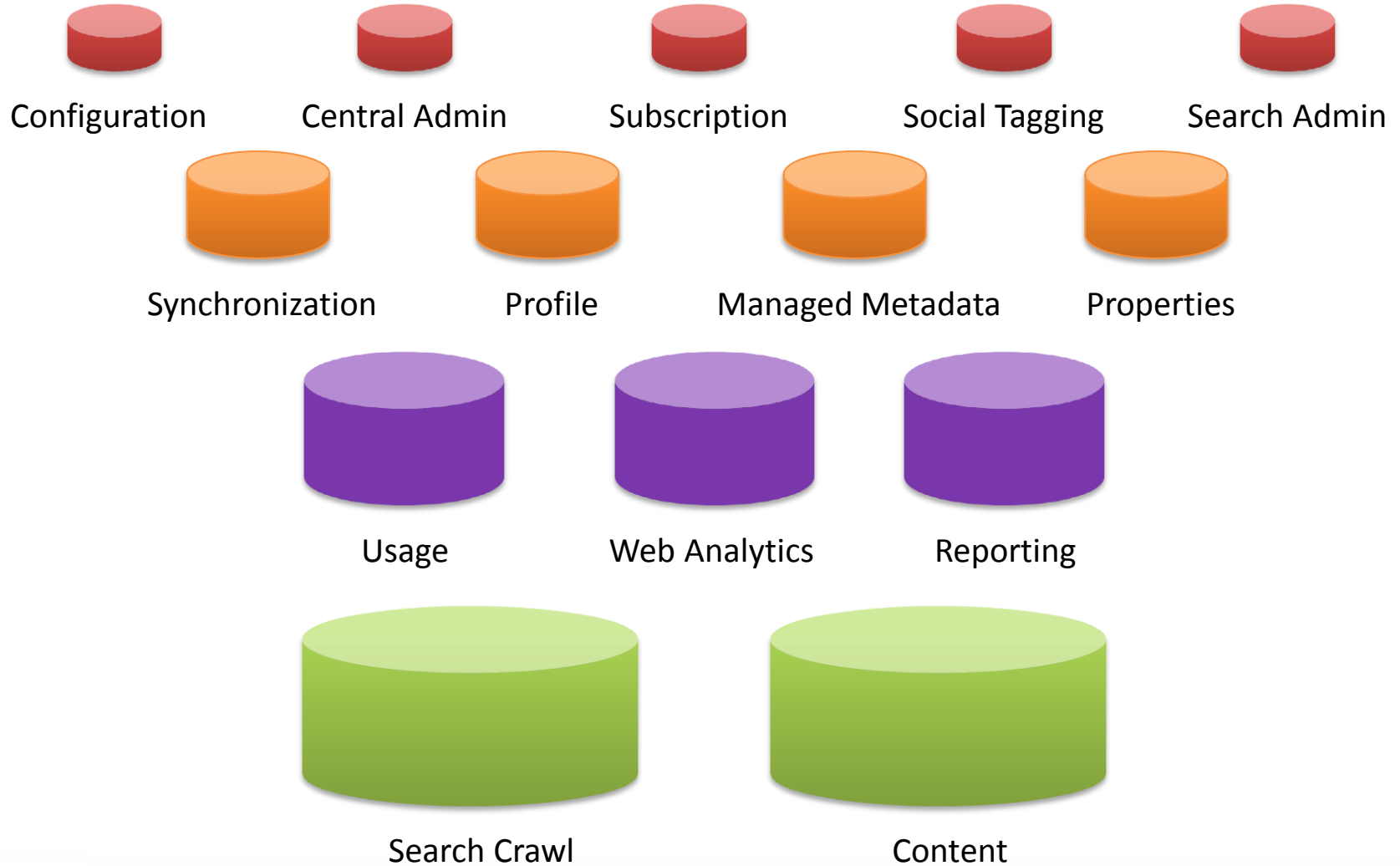
Database Servers

DATABASES

Database Operations



Database Types



Database Calculations



Variable	Value
# Documents	1,000,000
Average Size	150 KB
# List Items	3,000,000
# Versions	3

Total Database Sizing Estimate

864.4 GB

Formula: Database size = $((D \times V) \times S) + (10 \text{ KB} \times (L + (V \times D)))$

Database Size Estimates	
Content DB Size	486.5 GB
Crawl	22.4 GB
Property	7.3 GB
Profile	48.8 GB
Sync	30 GB
All other DB's	269.5 GB

Number of User Profiles

50,000

Content Databases



- Practical limit is 100GB
 - Max supported limit is 200GB
- Create separate databases for:
 - Site collections with large lists
 - Large numbers of subsites
 - Intensive read/write operations
 - Data isolation (security)
- Consider amount of time it takes to backup/restore

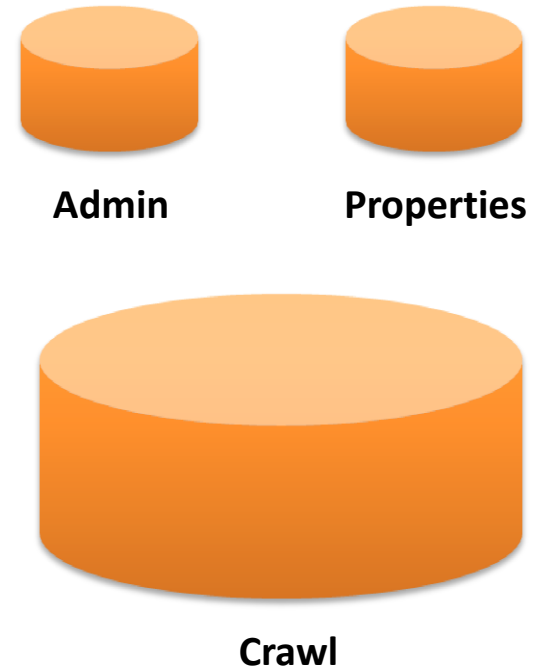


Content

Search Databases



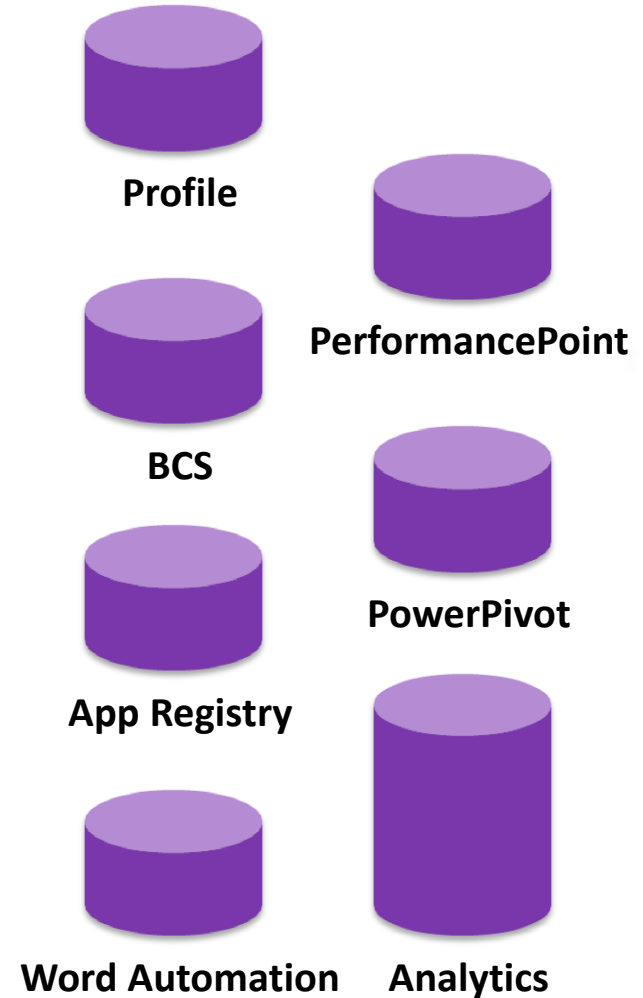
- Crawl databases can be extremely large
- High index sensitivity
- Heavy transactional volume
- Isolate crawl and temp databases
 - Distribute across spindles and LUN's
- Highest performance disk



Application Databases



- Small to moderate size
- Moderate transactional volume
- Group on moderate cost/performance disk
- **Analytics**
 - May be quite large
 - May require isolation
 - Reporting increases operational overhead



Database Management



- Manually configure auto-growth settings
- Defragment indexes
- Limit content database size per site collection
- Implement regular backup schedule to reduce log file size
- Isolate search databases
- Enforce quotas

PAGES

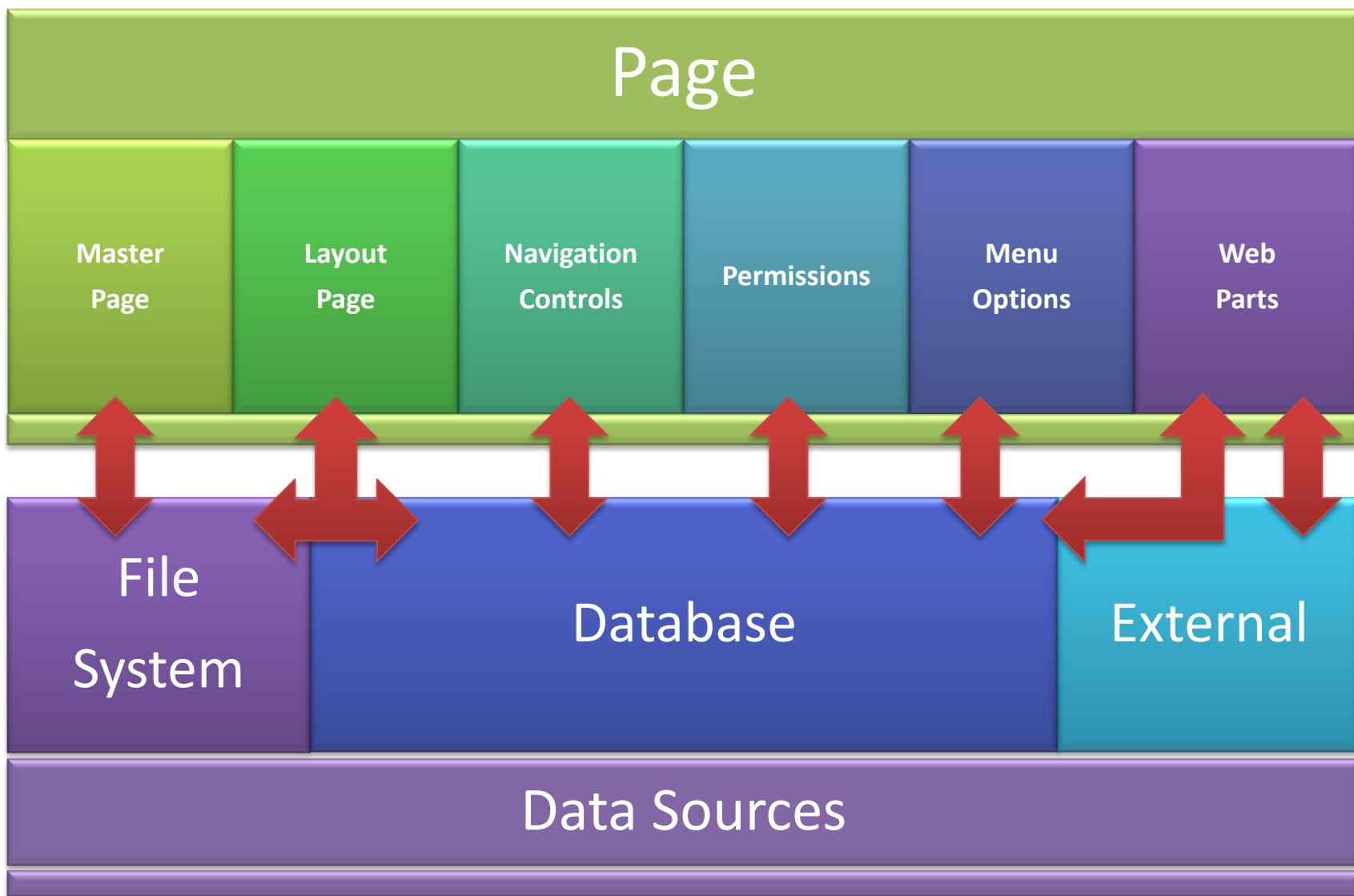
Page Controls



- Navigation
- Menus
- Ribbon
- Delegate
- Security Trimming
- Publishing Fields
- Search
- Layout
- Hidden



Page Data Queries



DEMO

Measuring Page Component Performance

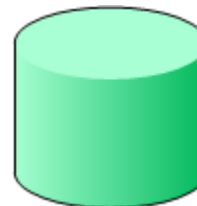
Page Customizations



Uncustomized Page

Customized Page

Content
stored on
disk, static
elements
cached,
assemblies
loaded

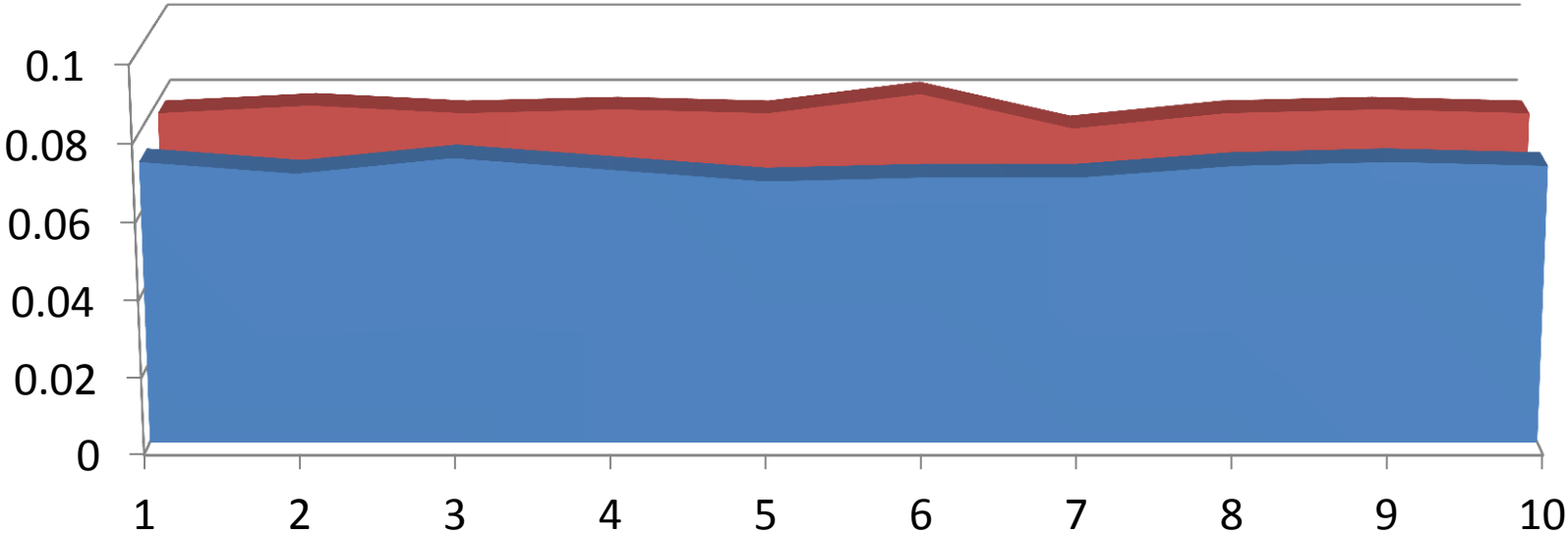


Content
retrieved
from
database on
each request

Test Results



Average Performance Delta: ~10%



Uncustomized

Customized

SharePoint Caching



Page

First request served from content database, output written to memory

Subsequent requests for same resource read from memory

Disk

File-system objects cached by IIS

Database objects not cached

Object

Commonly requested objects stored in memory

Cross-site queries cached in memory

IIS Compression



- Reduces size of files transmitted across the wire
- Configurable for various file types
- Increases CPU utilization on WFE's
- Does not effect dynamic content retrieved from database



DEMO

Effect of IIS Compression on Page Components

CSS, Images and HTML



- Start with a minimal master page
- Use CSS overrides – do not modify core CSS files!
- Minify and consolidate CSS files
- Use image stitching (CSS sprites) on pages with a lot of small images to reduce number of requests
- Store resources (style sheets, master pages, layout pages, images) on the PHYSICAL file system (i.e. /_layouts/) not the VIRTUAL file system (Style Library, Publishing Images)
 - Assets in libraries are stored in database
 - Easy for users to modify but reduce performance

Javascript and JQuery



- Load only necessary javascript for target users
 - Anonymous users don't need the same javascript references as contributors and designers
 - <http://www.sharepointnutsandbolts.com/2011/01/eliminating-large-js-files-to-optimize.html>
- Minify and consolidate javascript files
- JQuery is a tool – use it appropriately but don't abuse it
 - A page isn't "loaded" until ALL content is displayed
 - Many operations iterate recursively through the DOM
 - Simple syntax != increased performance
- Client Object Model

LISTS

Large Lists



- Just because a list CAN hold millions of items doesn't mean it SHOULD
- All user content in all lists throughout entire site collection is stored in a single table in the content database
- Folders improve view performance NOT query performance
- List view web parts are now XSLT based; however, large list displays may still require custom code

List Definitions



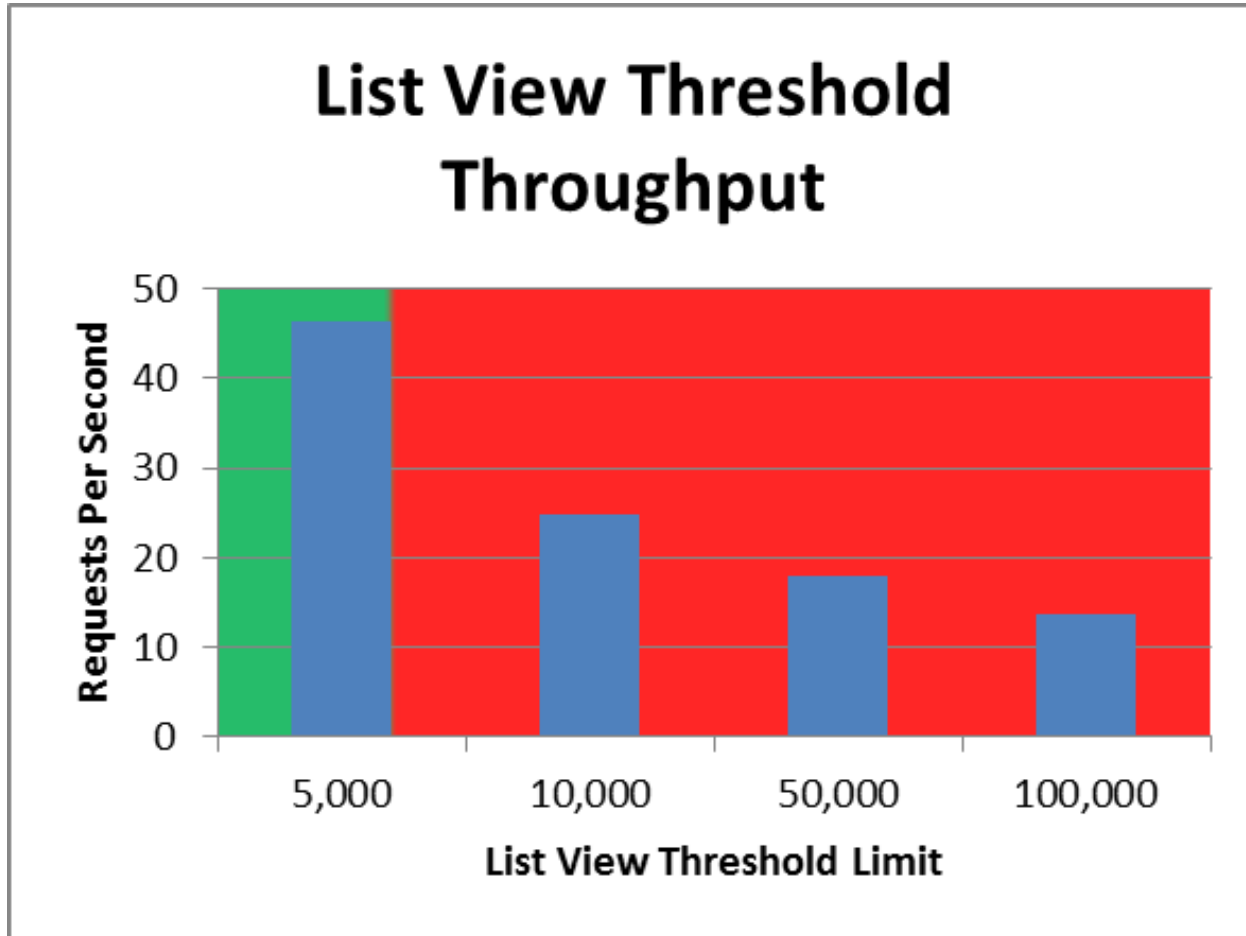
- Rows
 - More than 5,000 rows in a list is “large”
 - Index the columns
 - Use search
- Columns
 - More columns = more SQL rows
 - More SQL rows will slow down performance up to 35%
- Storage
 - Use Remote Blob Storage to mitigate large document libraries

Throttling and Locks



- SQL Server escalates row locks to table locks (> 5000)
- Query throttling reduces the impact of any single request by limiting the amount of data queried
- Throttling is configurable and can be altered for administrators and specific time periods
- Bit rate throttling controls download speeds of large objects (video, Flash, Silverlight)
 - Dependent upon BLOB cache

List Performance



COMPONENTS

Page Caching



- Store query results in page cache to reduce database calls
 - First load populates cache, subsequent loads read from cache
- Set cache expiration policies based on frequency of data modifications
 - Static content = long expiration window
 - Dynamic content = short expiration window
- Cache as much content as possible but be aware of impact on server resources

DEMO

Caching SharePoint Data Queries

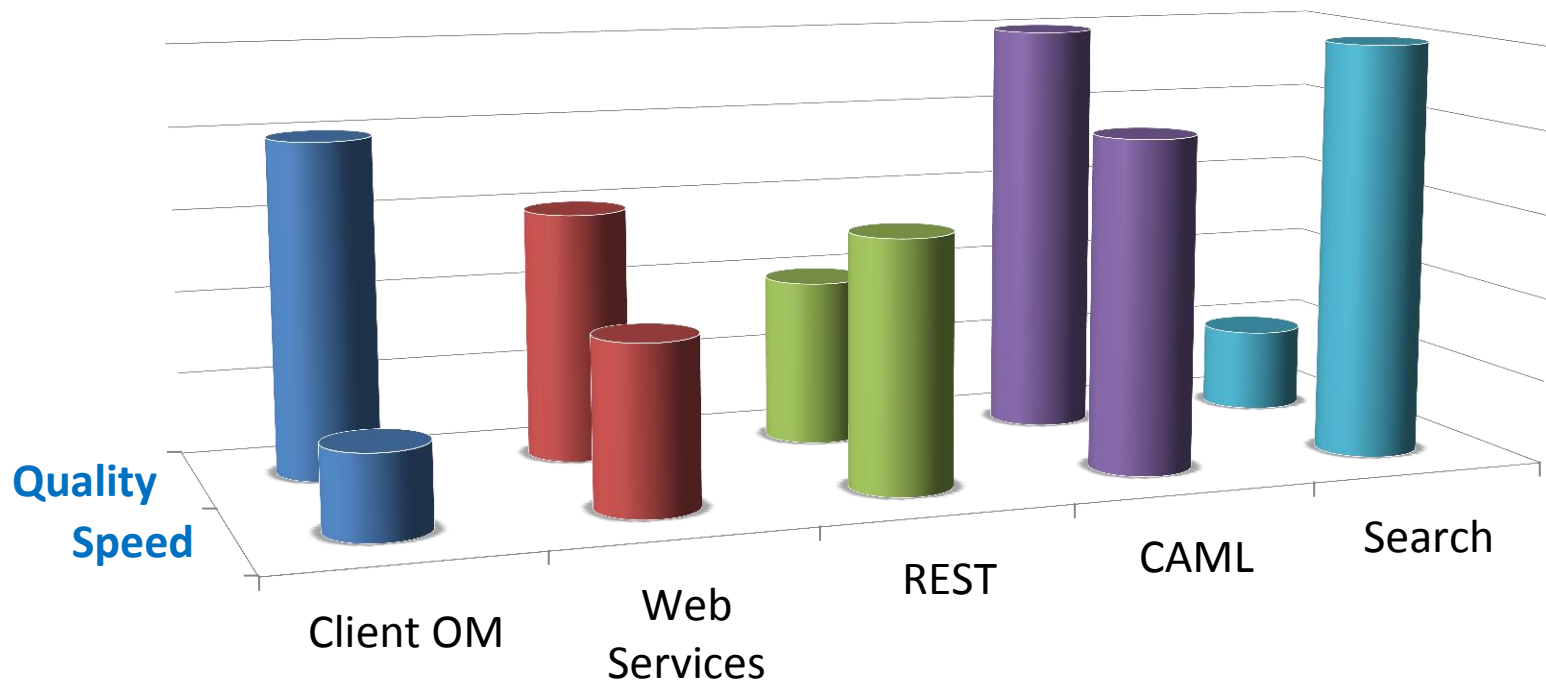
Data Access



- Server Object Model
- Client Object Model
- CAML (LINQ)
- PortalSiteMapProvider
- Web Services
- Search
- REST



The Need for Speed



Managing Large Data Sets



- Leverage search for mass data retrieval
- Use CAML queries for targeted item selection
 - Joins reduce performance
- Maintain SQL indexes for optimal query execution
- Manipulate item data using in-memory objects



- **SPSiteDataQuery**
 - Allows data to be retrieved across multiple list objects.
 - Provides moderate scalability but has performance implications for large datasets.
- **LINQ to SharePoint**
 - Does not work with external (BCS) lists.
 - Can result in suboptimal queries—test for performance.
- **Search**
 - Data is indexed and not real time (only valid as of last indexing operation).
 - Does not provide context of results (no SPListItem, SPList, SPWeb, etc.).
- **PortalSiteMapProvider**
 - Cached, non-real-time data.

Developer Dashboard



- Developer Dashboard provides metrics on object execution for individual pages
- Displays code-level request data for events
- Includes related database queries
- Identifies request allocations and control event offsets
- Used by developers to isolate performance issues

Dashboard Example



Developer Dashboard

- Request (GET:http://win2k8r2vm:80/SitePages/Home.aspx) (96.24 ms)
 - BeginRequestHandler (0.05 ms)
 - PostAuthenticateRequestHandler (0.00 ms)
 - PostResolveRequestCacheHandler (17.04 ms)
 - GetWebPartPageContent (15.19 ms)**
 - GetWebPartMetadataInfo (19.09 ms)
 - Wiki Edit OnInit (0.11 ms)
 - Wiki Edit OnInitComplete (0.64 ms)
 - Add WebParts (0.27 ms)
 - Contoso Performance List View with Search (0.02 ms)
 - Wiki Edit OnLoad (12.91 ms)
 - EnsureListItemsData (10.42 ms)
 - ToolBarMenuButton.CreateChildControls for SiteActions (0.32 ms)
 - ToolBarMenuButton.CreateChildControls for PersonalActions (0.11 ms)
 - SearchBoxEx.OnLoad (0.21 ms)
 - IsCheckedOutToSystem (0.01 ms)
 - SPPageStateControl:OnLoad (0.00 ms)
 - Activate web part connections (0.02 ms)
 - Wiki Edit OnPreRender (0.35 ms)
 - SPPageStateControl:OnPreRender (0.02 ms)
 - Wiki Edit OnPreRenderComplete (0.00 ms)
 - Wiki Edit Render (11.26 ms)
 - Render Ribbon. (2.05 ms)

Web Server

Execution Time	96.59 ms
Current User	WIN2K8R2VM\spadmin
Page Checkout Level	Published
Current SharePoint Operations	1
Log Correlation Id	d4d6265b-d824-4618-b60b-bc48604693f2

Asserts and Critical Events

Database Queries

proc_FetchDocForHttpGet	14.44 ms
SELECT t1.[TimeCreated]	5.52 ms
proc_EnumLists' CommandType:	31.72 ms

Service Calls

SPRequest Allocations

SPWeb: http://win2k8r2vm/SitePages/Home.aspx

WebPart Events Offsets

SPWebPartManager OnLoad	+0.00 ms
Contoso Performance List View with Search OnLoad	+0.02 ms
SPWebPartManager OnPreRender	+0.00 ms
Contoso Performance List View with Search OnPreRender	+32.30 ms

Control Events

Total Page Execution Time

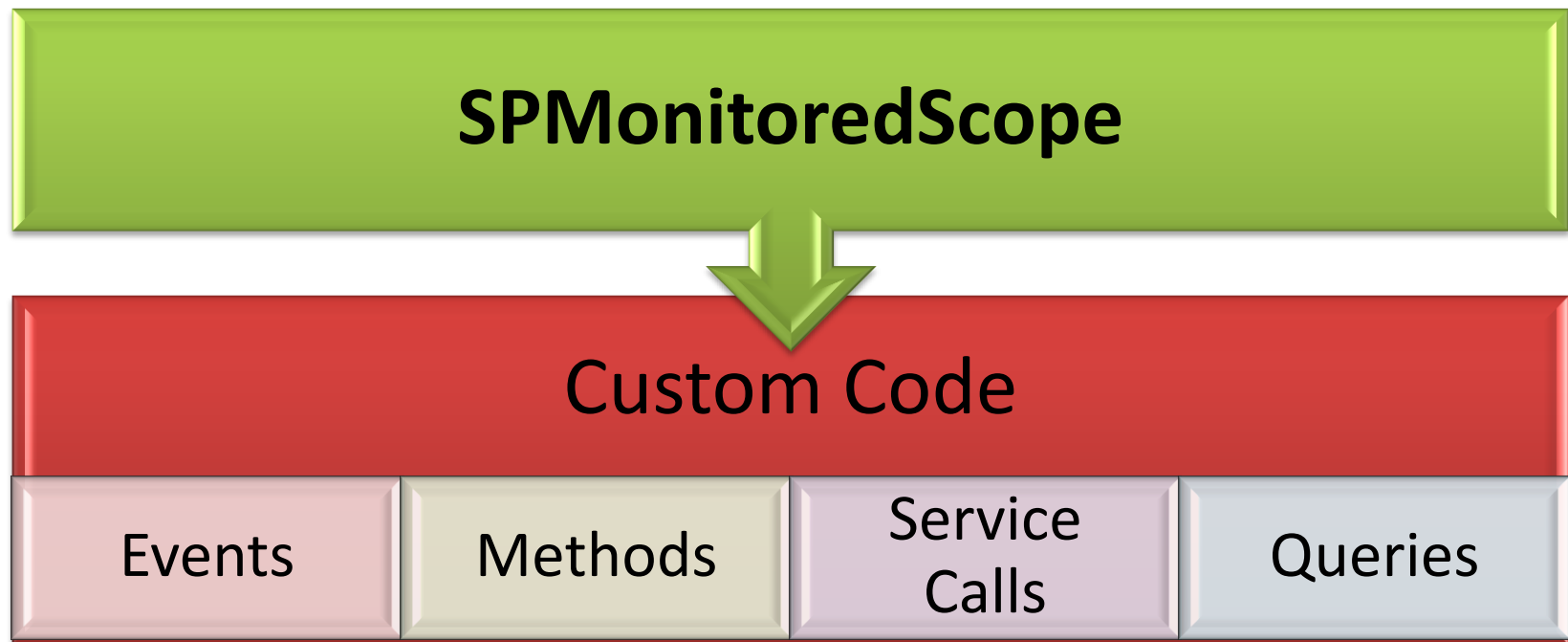
Related Queries

Show or hide additional tracing information . . .

Code Instrumentation



- `Microsoft.SharePoint.Utilities.SPMonitoredScope` provides instrumentation for custom code
- Subscopes are displayed as children of parent scope
- **Does not work with sandbox solutions**



DEMO

Instrumenting Code for the Developer Dashboard

Thanks to our Sponsors

