Code Solutions to Improve SharePoint Performance and Scalability via Caching

Sean P McDonough

sean@sharepointinterface.com

SharePoint *intersection*

Office 365 *intersection*

Code Solutions to Improve SharePoint Performance and Scalability via Caching

Sean P. McDonough
(@spmcdonough)
National Office 365 Solution Manager
Cardinal Solutions Group, Inc.

cardinal
PEOPLE.IDEAS.SOLUTIONS.

About Cardinal

Founded in 1996
Cincinnati Ohio

350+ FTEs
$50M+ Revenue

Cincinnati
Columbus
Charlotte
Raleigh
Tampa

Mobile
Portals & Collab
UXD
Application Dev
WEM
BI

Agile Coaching
Business Analysis
Project Management

cardinal

Session Overview

Quick Introduction

Component Caching Options

- ASP.NET Cache, AppFabric Caching

Caching for Controls

- Fragment Caching, Post-Cache Substitution

Caching for Pages

- IVaryByCustomHandler Implementation

Q&A Throughout

Why I care about caching

okay ...

Why I  caching

okay ...

Why I  caching

Formerly the architect for a
Fortune 25 company's publicly
facing SharePoint presence

okay ...

Why I  caching

Formerly the architect for a Fortune 25 company's publicly facing SharePoint presence

Highly trafficked environment with about 75,000 page views per hour (peak) in 2009

okay ...

Why I  caching

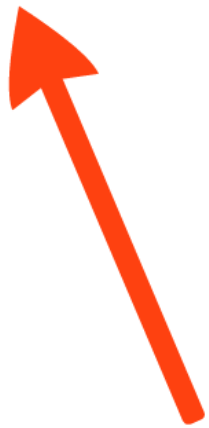
Formerly the architect for a Fortune 25 company's publicly facing SharePoint presence

Highly trafficked environment with about 75,000 page views per hour (peak) in 2009

Averaging (at peak) 1,000 requests/second into IIS

with about 73,000 page views per hour (peak) in 2009

k) 1,000 requests/second into IIS



Supported initially with just 2 web front ends (WFEs). Eventually moved to 4 WFEs for growth.

... and finally

... and finally

I'm sick and tired of hearing some people complain that "SharePoint doesn't scale"!!!

In my experience ...

SharePoint scaling and performance issues are more often than not due to poorly performing custom code

due to poorly performing custom code

What I'm about to share might help

Let's get rolling

First up:

Component-Level Caching

Dune Life

http://www.universe.com/Pages/DuneLife.aspx

Dune Life

? Sign In

Search this site...

Universe > Dune Life

Home

- Dune Life
- Giedi Prime Gazette
- Mentat Monitor
- Spice World

Welcome Back, Paul!

Today's Forecast

- Hot and Dry, 120F
- Slight Change of Worms

Right Now: **118F**

From Mentat News ...

- It is by will alone I set my mind in motion.
- It is by the juice of Sapho that thoughts acquire speed, the lips acquire stains, the stains become a warning.
- It is by will alone I set my mind in motion.

Spice Markets

(updated 15:23)

Dune Life

http://www.universe.com/Pages/DuneLife.aspx

Dune Life

? Sign In

Search this site...

Universe > Dune Life

Home

- Dune Life
- Giedi Prime Gazette
- Mentat Monitor
- Spice World

Welcome Back, Paul!

Today's Forecast

- Hot and Dry, 120F
- Slight Change of Worms

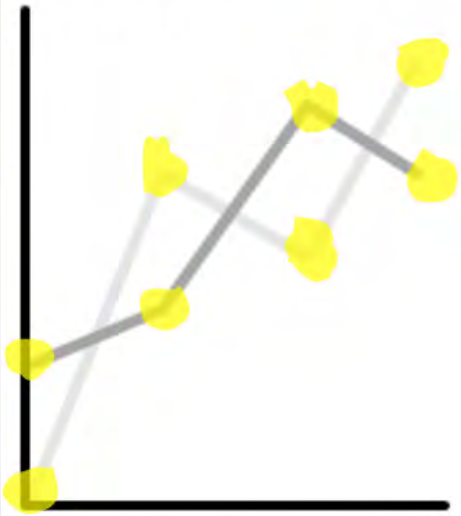
Right Now: **118F**

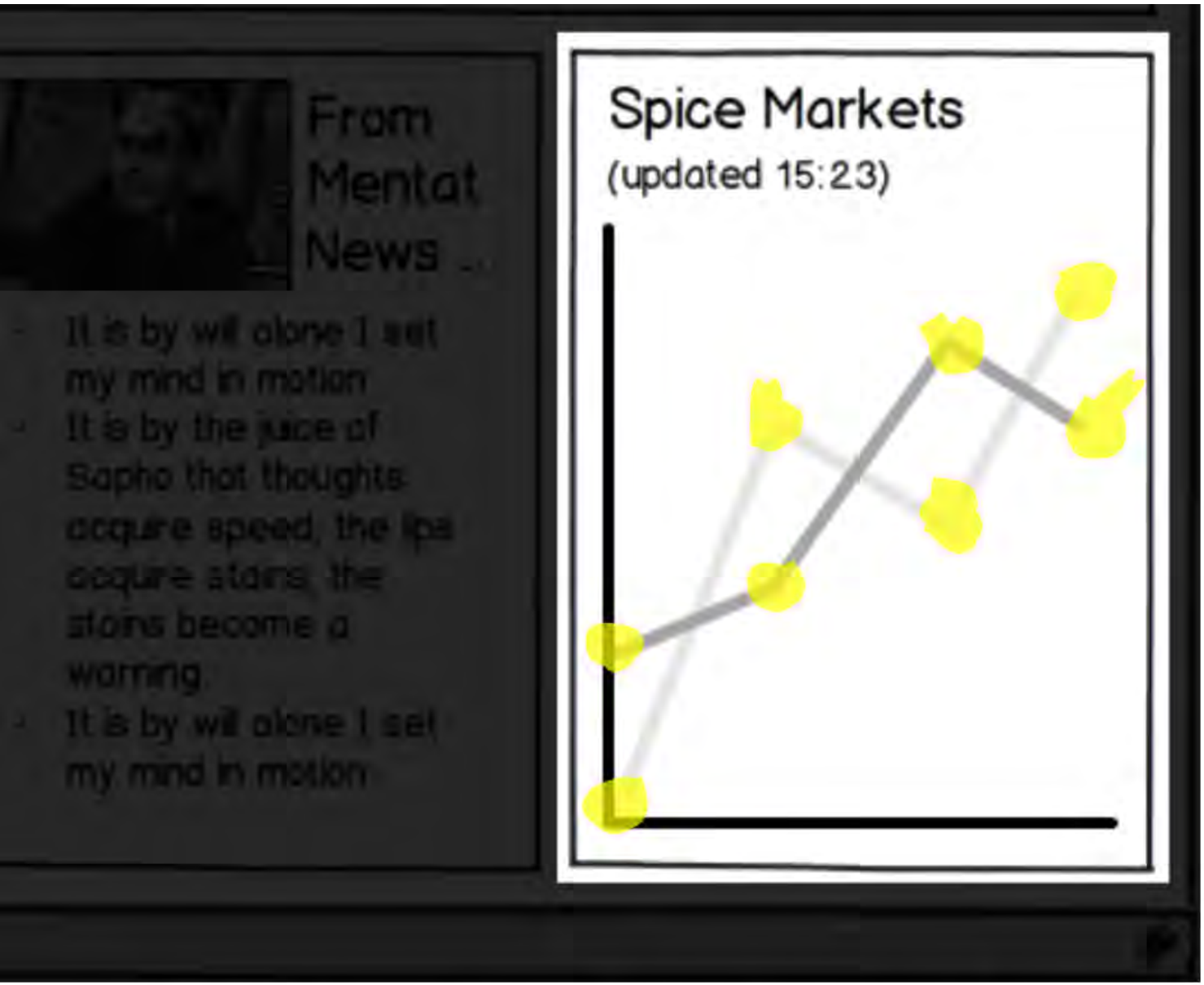
From Mentat News ...

- It is by will alone I set my mind in motion.
- It is by the juice of Sapho that thoughts acquire speed, the lips acquire stains, the stains become a warning.
- It is by will alone I set my mind in motion.

Spice Markets

(updated 15:23)





- Control rendering isn't complicated, but ...
- Data used is "expensive" (computation/latency)
- Need way to store expensive results between calls

- Control rendering isn't complicated, but ...
- Data used is "expensive" (computation/latency)
- Need way to store expensive results between calls

Two real options

ASP.NET Cache

AppFabric Cache

public sealed class **Cache**
Member of System.Web.Caching

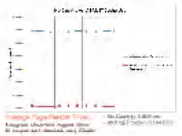
Summary:
Implements the cache for a Web application. This class cannot be inherited.

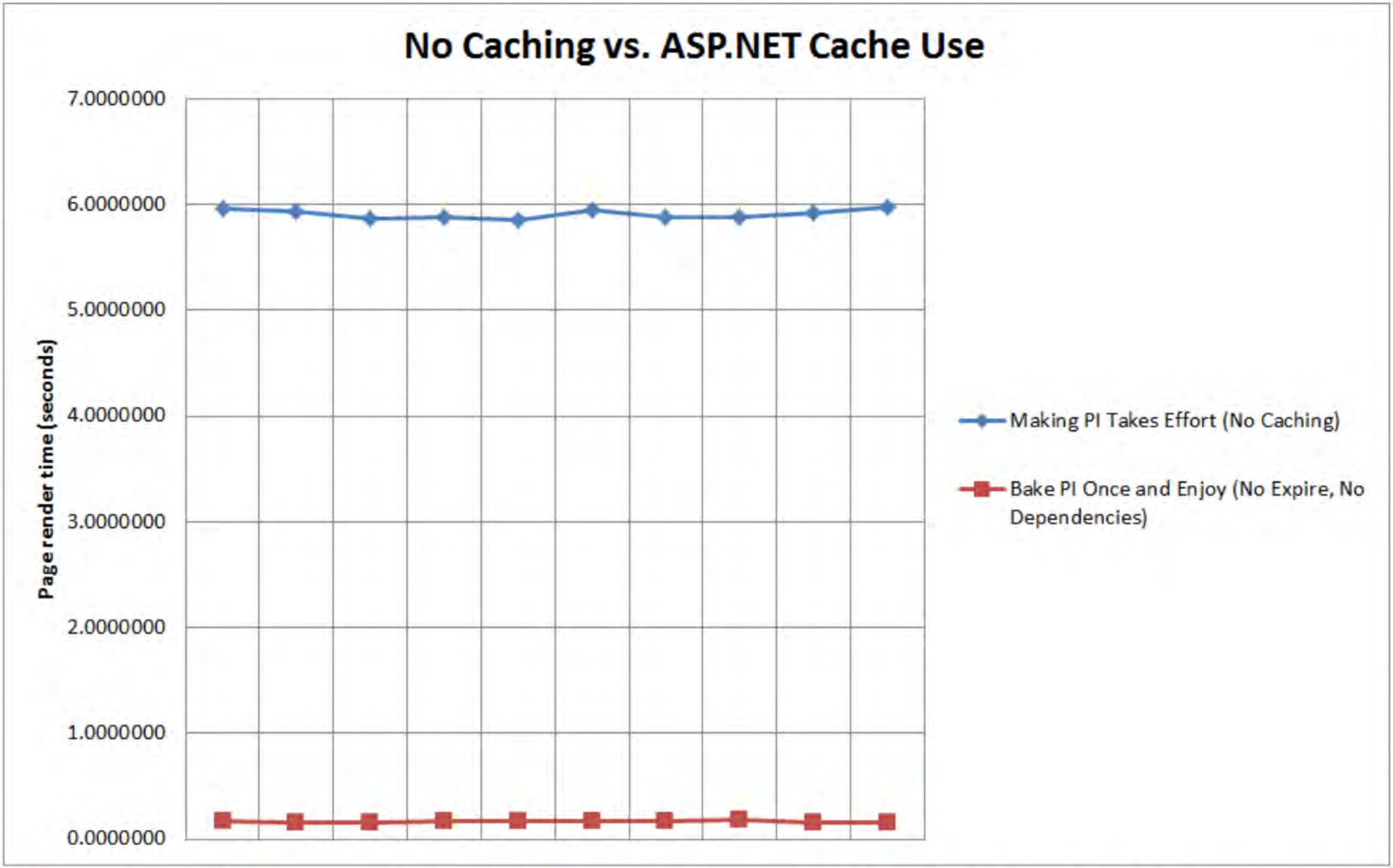
ASP.NET Cache

- System.Web.Caching.Cache class
- One instance per application domain
- Basically a key/value object dictionary
- In-memory use and thread-safe*
- Commonly accessed via Page and HttpContext objects
- Objects can be added with expiration windows, dependencies, & priority values
- Callbacks possible on object removal*

```
private String GetSomePi()
{
    // Attempt to retrieve a PI value from the ASP.NET Cache
    Object piValue = Cache[PI_VALUE_CACHE_KEY];

    // If the value isn't yet cached, compute it and cache it for later.
    if (piValue == null)
    {
        piValue = PiCalculator.Process(DIGITS_OF_PI_TO_COMPUTE);

        // Insert for indefinite time period
        Cache[PI_VALUE_CACHE_KEY] = piValue;

        //// Cache until a specific point in the future
        //Cache.Add(PI_VALUE_CACHE_KEY,
        //          piValue,
        //          null,
        //          DateTime.Now.AddSeconds(15),
        //          Cache.NoSlidingExpiration,
        //          CacheItemPriority.Normal,
        //          null);

        //// Cache for a sliding window of 3 seconds
        //Cache.Add(PI_VALUE_CACHE_KEY,
        //          piValue,
        //          null,
        //          Cache.NoAbsoluteExpiration,
        //          TimeSpan.FromSeconds(3),
        //          CacheItemPriority.Normal,
        //          null);
    }

    return piValue.ToString();
}
```

Demo





Average Page Render Times

Anonymous client-side request times;
10 samples each obtained using Fiddler

- No Caching: 5.909 sec
- ASP.NET Cache: 0.1641 sec

Limitations and Watch-Outs

- Not a durable store
- Don't assume something you put in will always be available
- Cache contents not available across WFEs in a load-balanced environment

Sum-up: Safe for general use. Just remember the cache is shared.

- Control rendering isn't complicated, but ...
- Data used is "expensive" (computation/latency)
- Need way to store expensive results between calls

Two real options

ASP.NET Cache

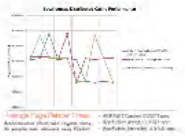
AppFabric Cache

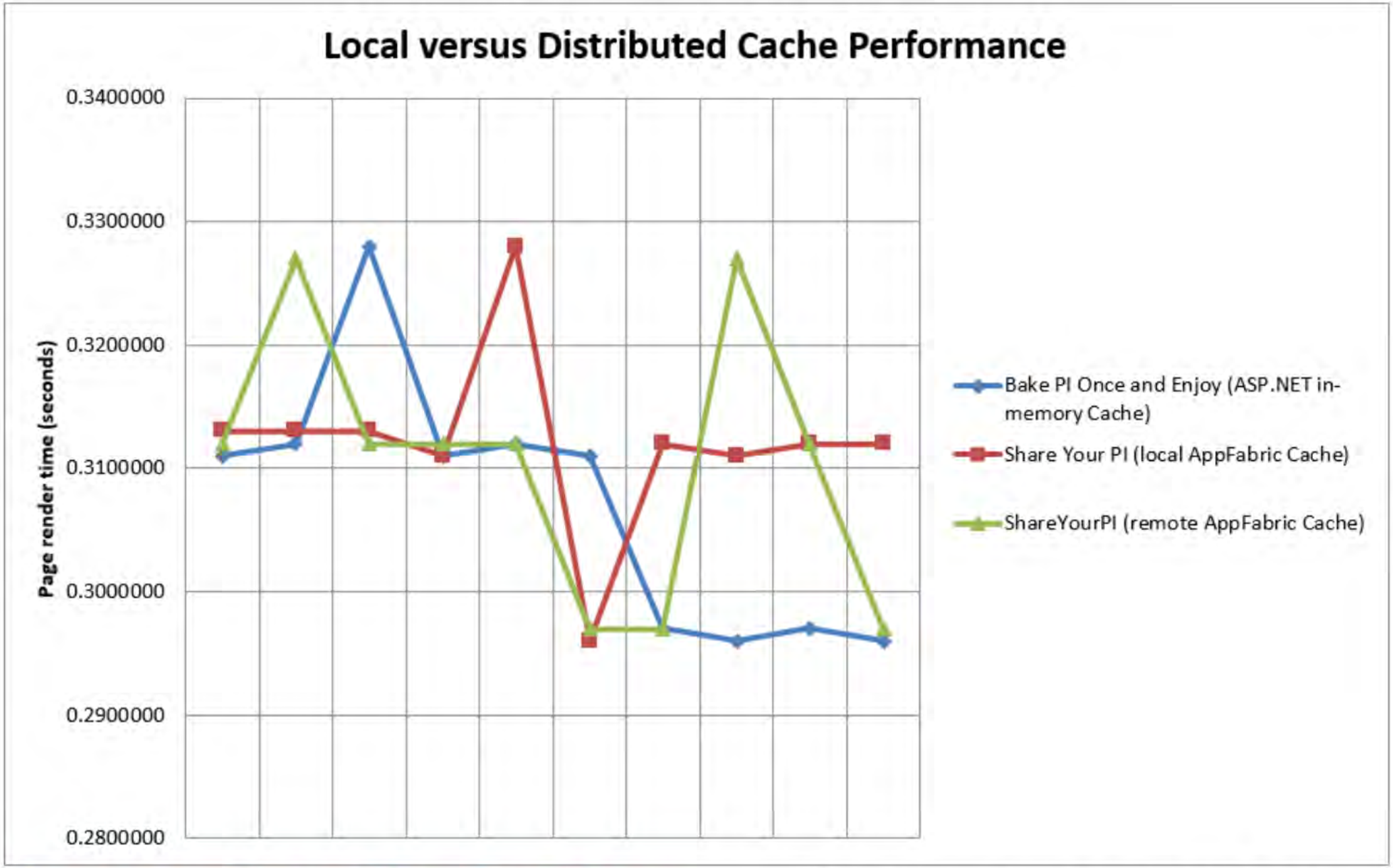
AppFabric Cache

- Microsoft AppFabric (1.1) for Windows Server
- Provides highly available distributed caching
- Exists independent of SharePoint (and thus will work with any version of SharePoint)
- From a code perspective, very similar to writing code for the ASP.NET (local) cache
- Requires significant external configuration and setup, so you'll want to become good friends with your SharePoint administrators

```
// Constants for use in interacting with the AppFabric cache
private const String CACHE_SERVER_NAME = "afhost";       // Separate AF Cache - take this approach
// private const String CACHE_SERVER_NAME = "localhost";       // SharePoint's AF Cache - DON'T DO THIS!
private const Int32 CACHE_SERVER_PORT = 22233;


// Private members to support singleton and cache interactions
private static DataCache _appFabricCacheInstance;


// Leverage static constructor to ensure one-time initialization for the
// cache instance.
0 references
static AppFabricSingleton()
{
    // Identify the cache host(s) to which we want to connect
    List<DataCacheServerEndpoint> cacheServers = new List<DataCacheServerEndpoint>();
    cacheServers.Add(new DataCacheServerEndpoint(CACHE_SERVER_NAME, CACHE_SERVER_PORT));
    DataCacheFactoryConfiguration factoryConfig = new DataCacheFactoryConfiguration();
    factoryConfig.Servers = cacheServers;

    // Without this setting, an SSPI error can be thrown when calling to an AppFabric Cache
    // Host that is configured to run under a domain account (instead of NETWORK SERVICE)
    factoryConfig.DataCacheServiceAccountType = DataCacheServiceAccountType.DomainAccount;

    // If failure occurs, it will likely be with the next line to establish the cacheFactory. If you
    // see HTTP 403 errors, your Windows Firewall may be blocking calls, you may not have granted
    // the necessary accounts access to the AppFabric Cache (e.g., calls to the AppFabric typically
    // come from the Application Pool's identity)
    DataCacheFactory cacheFactory = new DataCacheFactory(factoryConfig);
    _appFabricCacheInstance = cacheFactory.GetDefaultCache();
}
```

Demo





Average Page Render Times

Authenticated client-side request times;
10 samples each obtained using Fiddler

- ASP.NET Cache: 0.3071 sec
- AppFabric (local): 0.3121 sec
- AppFabric (remote): 0.3105 sec

Limitations and Watch-Outs

Straight from TechNet ...

Important:

If you are using custom applications in SharePoint Server 2013 which use the AppFabric client APIs, or are creating custom caches, you should create a separate AppFabric cache cluster to support your custom applications. Do not use the AppFabric cache cluster supporting your SharePoint Server 2013 farm. Run your separate AppFabric cache cluster for your custom applications on separate servers from the servers dedicated to your SharePoint Server 2013 farm.

- You will find code samples from people that use SharePoint's cache cluster
- Don't be tempted by the Dark Side; establish your own cache cluster on one or more non-SharePoint servers

Here there be monsters!

Coding for AppFabric is relatively easy.
Configuring AppFabric properly is not.

abric is relatively easy.

Fabric properly is not.

Cumulative updates can go a long way towards stabilizing your environment (and avoiding this)

Latest is CU6

Next-up:

Control-Level Caching

Dune Life

http://www.universe.com/Pages/DuneLife.aspx

Dune Life

? Sign In

Search this site...

Universe > Dune Life

Home

- Dune Life
- Giedi Prime Gazette
- Mentat Monitor
- Spice World

Welcome Back, Paul!

Today's Forecast

- Hot and Dry, 120F
- Slight Change of Worms

Right Now: **118F**

From Mentat News ...

- It is by will alone I set my mind in motion.
- It is by the juice of Sapho that thoughts acquire speed, the lips acquire stains, the stains become a warning.
- It is by will alone I set my mind in motion.

Spice Markets

(updated 15:23)


Dune Life
http://www.universe.com/Pages/DuneLife.aspx
Dune Life
?
Search this site...
Home
Dune Life
Giedi Prime Gazette
Mentat Monitor
Spice World
Universe > Dune Life
Welcome Back, Paul!
Today's Forecast
Hot and Dry, 120F
Slight Change of Worms
Right Now: 118F
From Mentat News ...
It is by will alone I set my mind in motion.
It is by the juice of Sapho that thoughts acquire speed, the lips acquire stains, the stains become a warning.
It is by will alone I set my mind in motion.
Spice Markets
(updated 15:23)

- Control displays static content or ...
- Entire HTML output block generated by control changes infrequently and/or according to predictable variables/patterns

Fragment Caching

An easily implemented way to cache the entire block of HTML that is generated by a control

To implement, simply add something like the following to an ASCX control file:

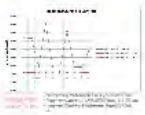
```
<%@ OutputCache Duration="120" VaryByParam="none" %>
```

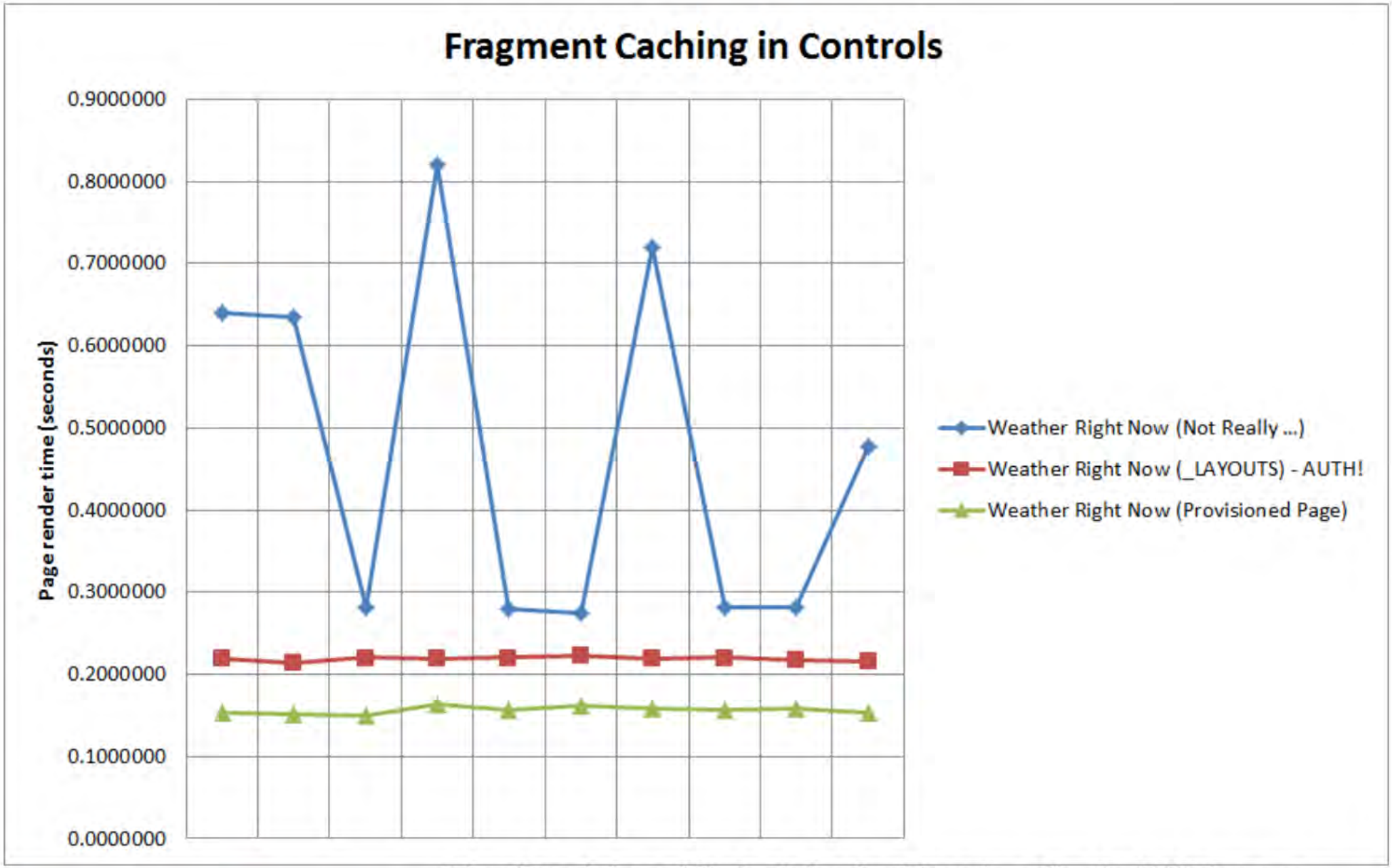
(Common) options to vary output exist based on:

- HTTP Header
- Query string value (GET) or parameter (POST)
- Value of child control in ASCX

```
<%@ Assembly Name="$SharePoint.Project.AssemblyFullName$" %>
<%@ Assembly Name="Microsoft.Web.CommandUI, Version=14.0.0.0, Culture=neutral, PublicKeyToken=71e9bce11
<%@ Register Tagprefix="SharePoint" Namespace="Microsoft.SharePoint.WebControls" Assembly="Microsoft.Sh
<%@ Register Tagprefix="Utilities" Namespace="Microsoft.SharePoint.Utilities" Assembly="Microsoft.Share
<%@ Register Tagprefix="asp" Namespace="System.Web.UI" Assembly="System.Web.Extensions, Version=3.5.0.0
<%@ Import Namespace="Microsoft.SharePoint" %>
<%@ Register Tagprefix="WebPartPages" Namespace="Microsoft.SharePoint.WebPartPages" Assembly="Microsoft
<%@ Control Language="C#" AutoEventWireup="true" CodeBehind="WeatherRightNowScraper.ascx.cs" Inherits="

<%@ OutputCache Duration="120" VaryByControl="ZipCodeTextbox" %>

<asp:Panel ID="MainPanel" runat="server" BorderStyle="Solid" BorderWidth="1px" style="padding: 5px;">
    TIME WHEN CONTROL WAS RENDERED: <asp:Label ID="GenerationTimeLabel" runat="server"></asp:Label>
    <br />
    TIME TO GENERATE HTML OUPUT: <asp:Label ID="TimeToComputeLabel" runat="server"></asp:Label>
    <hr />
    ZIP CODE: <asp:TextBox ID="ZipCodeTextbox" runat="server">45244</asp:TextBox>
    <input id="SubmitButton" type="submit" value="Get Weather" />
    <br />
    <asp:Literal ID="WeatherLiteral" runat="server"></asp:Literal>
</asp:Panel>
```

Demo





Average Page Render Times

- No Caching (Safe Mode Parsing*): 0.4691 sec
- Fragment Caching (_LAYOUTS Page): 0.2187 sec
- Fragment Caching (Provisioned Page): 0.1566 sec

Limitations and Watch-Outs

- Test your VaryBy... parameter settings carefully
- If using both page-level and control-level caching, page-level will trump control-level (duration) settings
- If caching doesn't appear to work, consider that the safe mode parser may be engaged. Work around it with a provisioned page, _layouts page, or another (safe) alternative

Sum-up: Safe way to cache control content that changes infrequently

Dune Life

http://www.universe.com/Pages/DuneLife.aspx

Dune Life

? Sign In

Search this site...

Universe > Dune Life

Home

- Dune Life
- Giedi Prime Gazette
- Mentat Monitor
- Spice World

Welcome Back, Paul!

Today's Forecast

- Hot and Dry, 120F
- Slight Change of Worms

Right Now: **118F**

From Mentat News ...

- It is by will alone I set my mind in motion.
- It is by the juice of Sapho that thoughts acquire speed, the lips acquire stains, the stains become a warning.
- It is by will alone I set my mind in motion.

Spice Markets

(updated 15:23)

Dune Life

http://www.universe.com/Pages/DuneLife.aspx

Dune Life

? Sign In

Search this site...

Universe > Dune Life

Home

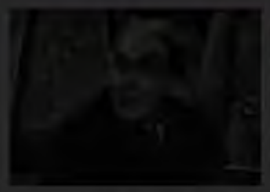
- Dune Life
- Giedi Prime Gazette
- Mentat Monitor
- Spice World

Welcome Back, Paul!

Today's Forecast

- Hot and Dry, 120F
- Slight Change of Worms

Right Now: 118F



From Mentat News ...

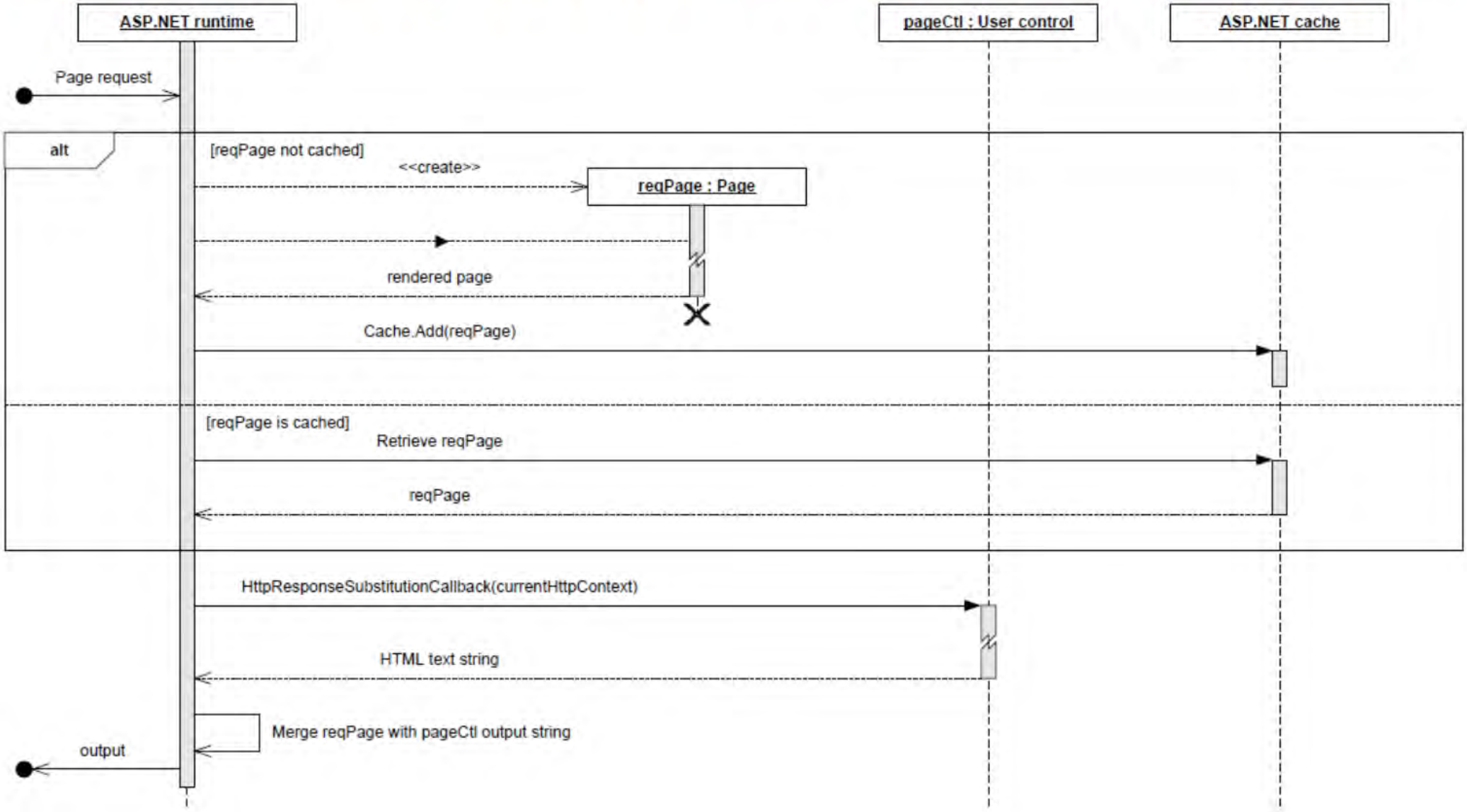
- It is by will alone I set my mind in motion.
- It is by the juice of Sapho that thoughts acquire speed, the lips acquire stains, the stains become a warning.
- It is by will alone I set my mind in motion.

Spice Markets

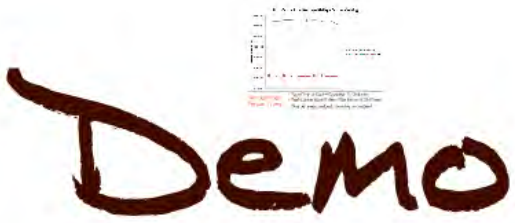
(updated 15:23)

- You are leveraging page output caching (i.e., the entire page's HTML output gets cached)
- Your control contains a mix of static and dynamic content
- You need a way to update the dynamic part (e.g., the "Right Now" temperature)

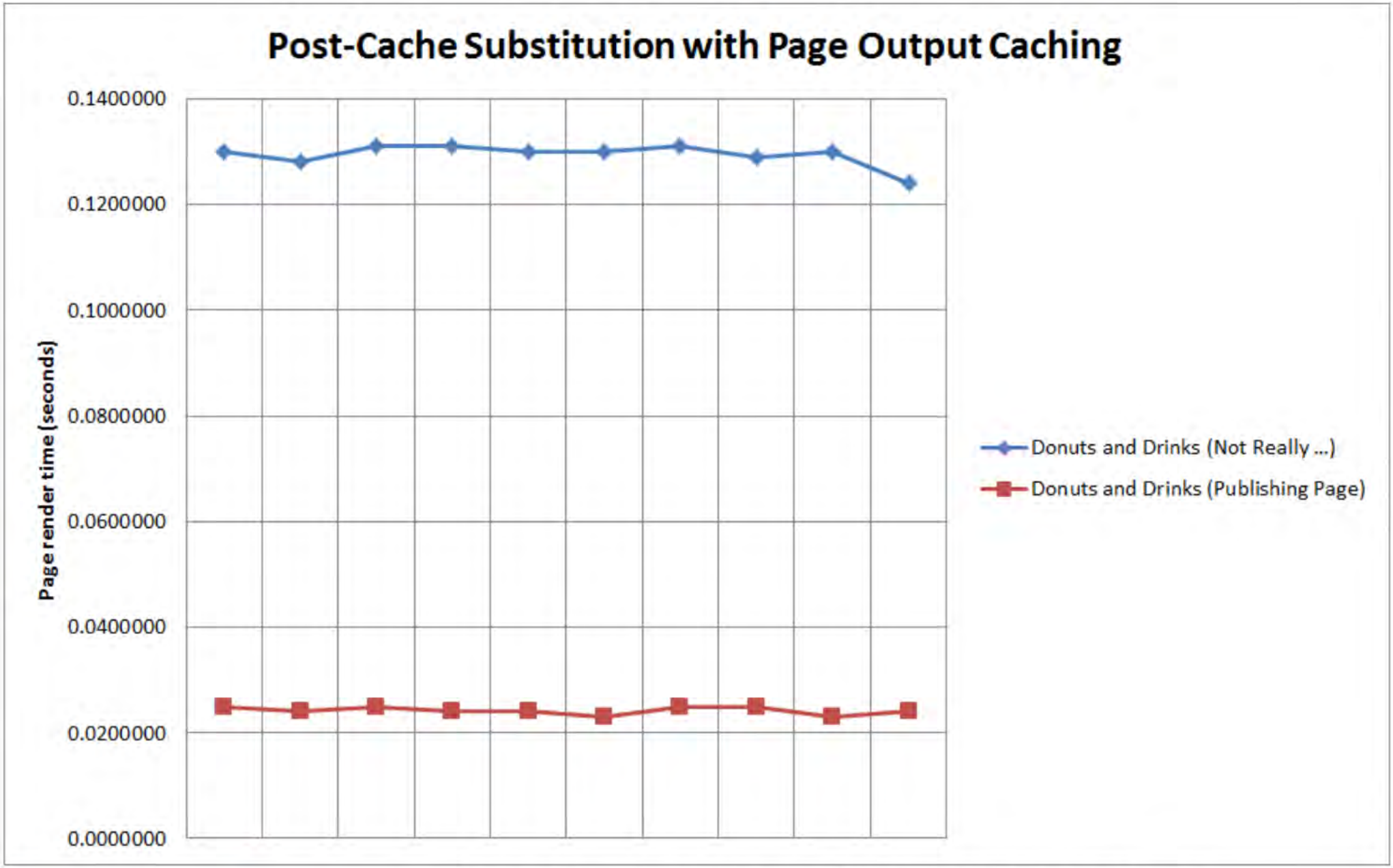
Post-Cache Substitution



Page output caching "with benefits"




```
<asp:Panel ID="MainPanel" runat="server">
    <div style="text-align: center">
        <h2>Donut Caching: Now with Beverage!</h2>
    </div>
    <div style="position:relative;">
        <div style="float:left; width:50%; text-align:center;">
            <img alt="Have a yummy donut!" src="../../_layouts/images/CcsExamples/PCS_Donut.jpg"/>
            <table style="font-family: Arial, Helvetica, sans-serif; text-align:center; width:100%">
                <tr>
                    <td>
                        Enjoy a tasty donut and ...
                    </td>
                </tr>
                <tr>
                    <td>
                        Prepared at <asp:Label runat="server" ID="DonutPreparedLabel"></asp:Label>
                    </td>
                </tr>
            </table>
        </div>
        <asp:Substitution runat="server" ID="BeverageSubstitution" MethodName="GetBeverageHtmlBlock" />
    </div>
</asp:Panel>
```



Average Page Render Times

- Page Output Cache Disabled: 0.1294 sec
- Post-Cache Substitution (Pub Page): 0.0240 sec

This is page output caching in action!

Limitations and Watch-Outs

- Remember that page output caching needs to be enabled to actually make this work
- If caching isn't working at all, use the Debug Cache Information option to determine if the host page is being output cached
- Obscure issue: if you override rendering at the page level (e.g., within the master page), post-cache substitution will break

Sum-up: Great complement to page output caching for controls that contain some dynamic content

Heading into home:

Customing Page Output Caching

Dune Life

http://www.universe.com/Pages/DuneLife.aspx

Dune Life

? Sign In

Search this site...

Universe > Dune Life

Home

- Dune Life
- Giedi Prime Gazette
- Mentat Monitor
- Spice World

Welcome Back, Paul!

Today's Forecast

- Hot and Dry, 120F
- Slight Change of Worms

Right Now: **118F**

From Mentat News ...

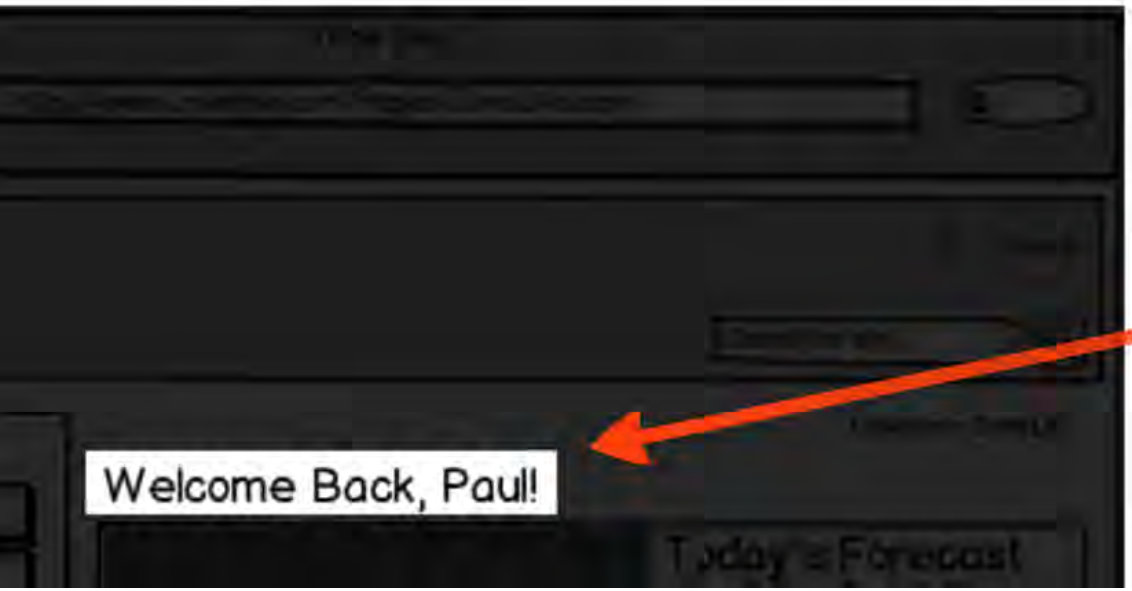
- It is by will alone I set my mind in motion.
- It is by the juice of Sapho that thoughts acquire speed, the lips acquire stains, the stains become a warning.
- It is by will alone I set my mind in motion.

Spice Markets

(updated 15:23)


Dune Life
http://www.universe.com/Pages/DuneLife.aspx
Dune Life
?
Sign In
Search this site...
Universe > Dune Life
Home
Dune Life
Giedi Prime Gazette
Mentat Monitor
Spice World
Welcome Back, Paul!
Today's Forecast
- Hot and Dry, 120F
- Slight Change of Worms
Right Now: 118F
From Mentat News ...
- It is by will alone I set my mind in motion.
- It is by the juice of Sapho that thoughts acquire speed, the lips acquire stains, the stains become a warning.
- It is by will alone I set my mind in motion.
Spice Markets
(updated 15:23)

- You need a way to more granularly control SharePoint's page output caching, or ...
- You need a way to control or completely disable caching across site collections based on run-time conditions/circumstances, or ...
- You want to affect output caching changes through SharePoint plumbing (w/o controls)



Conditionally include or exclude full page from page output cache

IVaryByCustomHandler

- Exposes one method for our use: the GetVaryByCustomString method
- Method gets called during ResolveRequestCache and UpdateRequestCache event stages
- You supply a return string that gets built into the key that is used to partition pages in the cache.
- You have additional levels of control, such as the ability to disable output caching.

Implementation Process

- Create a class that derives from SPHttpApplication and implements both IHttpModule and IVaryByCustomHandler*
- Register the derived class for notifications using RegisterGetVaryByCustomStringHandler
- Build detection & caching logic into the GetVaryByCustomString method*
- Use a FeatureReceiver to register the class as an HttpModule with help from the SPWebConfigModification type

Ignore the MSDN sample directing you to modify the Global.asax file

Implementation Process

- Create a class that derives from SPHttpApplication and implements both IHttpModule and IVaryByCustomHandler*
- Register the derived class for notifications using RegisterGetVaryByCustomStringHandler
- Build detection & caching logic into the GetVaryByCustomString method*
- Use a FeatureReceiver to register the class as an HttpModule with help from the SPWebConfigModification type

Perform ACL Check	No
Enabled	Yes
Duration	120
Check for Changes	No
Vary by Custom Parameter	Browser, HadACookieCustomCaching
Vary by HTTP Header	
Vary by Query String Parameters	
Vary by User Rights	No

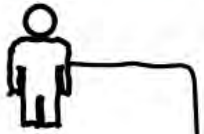
"Vary by Custom Parameter" in your output cache profile activates the handler!

Implementation Process

- Create a class that derives from SPHttpApplication and implements both IHttpModule and IVaryByCustomHandler*
- Register the derived class for notifications using RegisterGetVaryByCustomStringHandler
- Build detection & caching logic into the GetVaryByCustomString method*
- Use a FeatureReceiver to register the class as an HttpModule with help from the SPWebConfigModification type

Application Setup

Assumption: application pool isn't spun-up yet.



1. A user requests a page from SharePoint (e.g., default.aspx)

5. Our custom caching provider is instantiated, and a reference to the SharePoint SPHttpApplication is passed in.

Q: How does SharePoint know about the custom caching module?

A: The caching module (an HttpModule) is wired into the ASP.NET pipeline using SPWebConfigModification entries in the FeatureActivated event of a FeatureReceiver

A section from the web.config

instantiate custom handler

ASP.NET Worker Process

instantiate SharePoint application

2. ASP.NET spins-up a worker process and an application pool (as needed) for SharePoint.

3. The web.config the entities it references (e.g., HttpHandlers and HttpModules) are processed for initialization.

4. An instance of the SharePoint application (SPHttpApplication) is started-up to process SharePoint requests (for example, our page request)

SharePoint Application

RegisterGetVaryByCustomStringHandler()

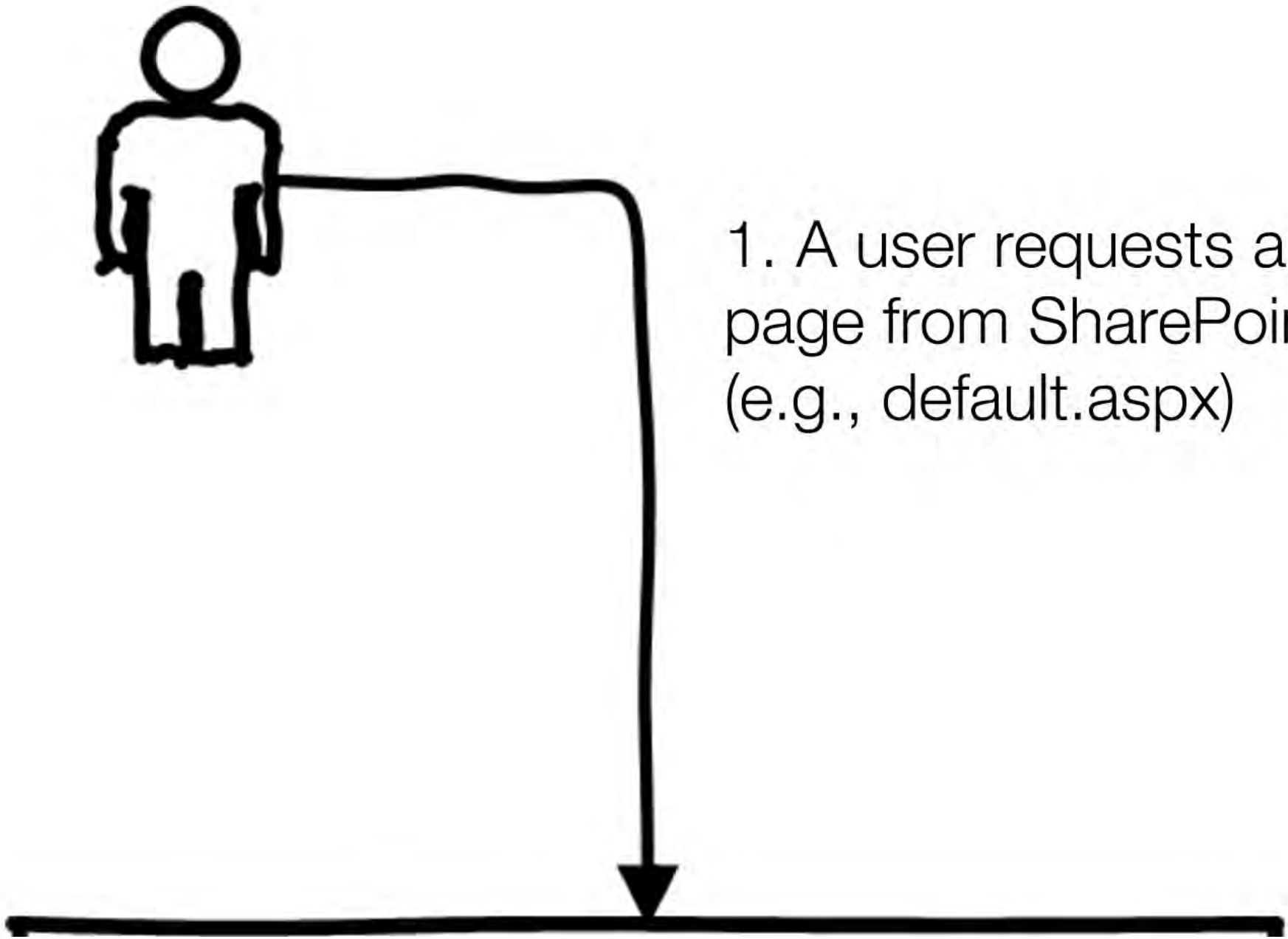
IHttpModule.Init()

Custom Caching Provider

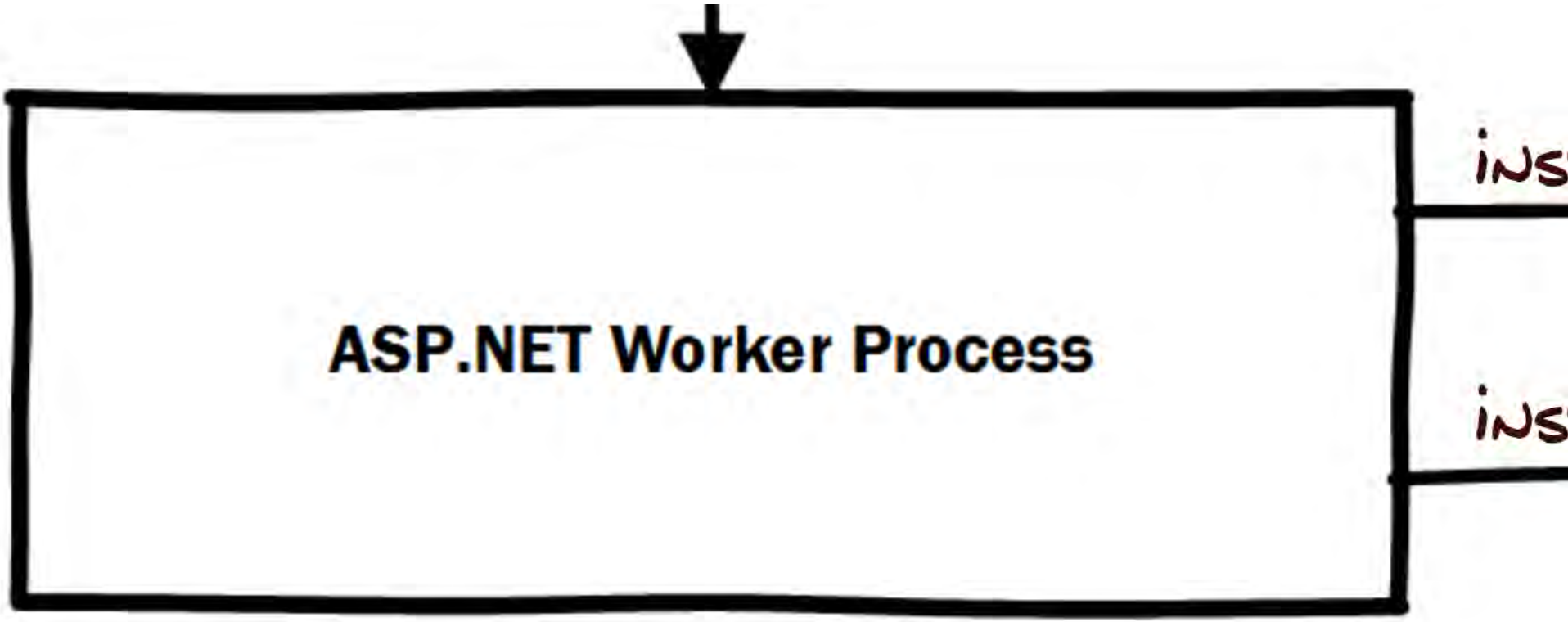
7. When the custom caching provider calls the RegisterGetVaryByCustomStringHandler() method and passes a reference to itself, SharePoint knows to call the provider anytime a page output request is being handled.

6. In the Init() method, the custom caching provider gets a reference to the SharePoint application. It uses that reference to call into SharePoint to register

Assumption: application pool isn't spun-up yet.



1. A user requests a page from SharePoint (e.g., default.aspx)

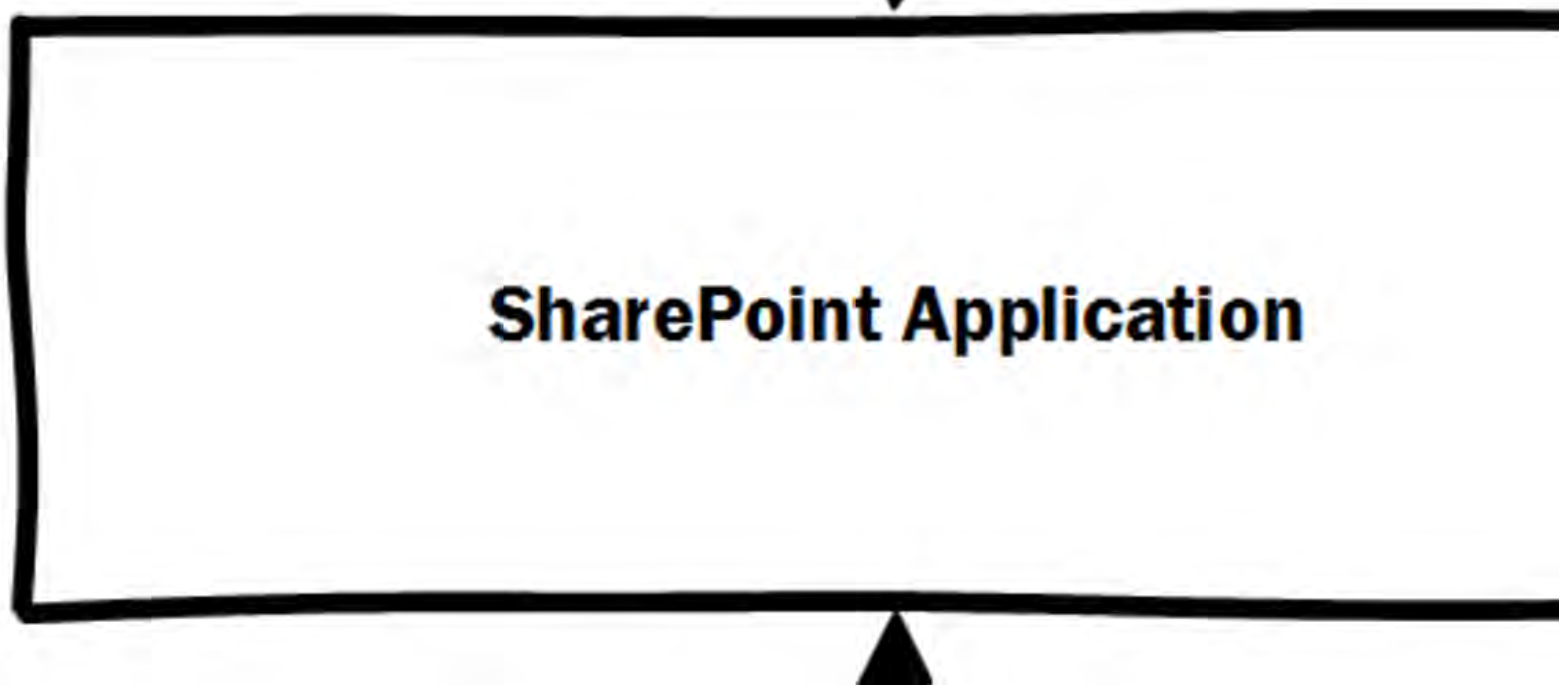


2. ASP.NET spins-up a worker process and an application pool (as needed) for SharePoint.

3. The web.config the entities it references (e.g., HttpHandlers and HttpModules) are processed for initialization.

instantiate SharePoint application

4. An instance of the SharePoint application (SPHttpApplication) is started-up to process SharePoint requests (for example, our page request)



5. Our custom caching provider is instantiated, and a reference to the SharePoint SPHttpApplication is passed in.

Q: How does SharePoint know about the custom caching module?

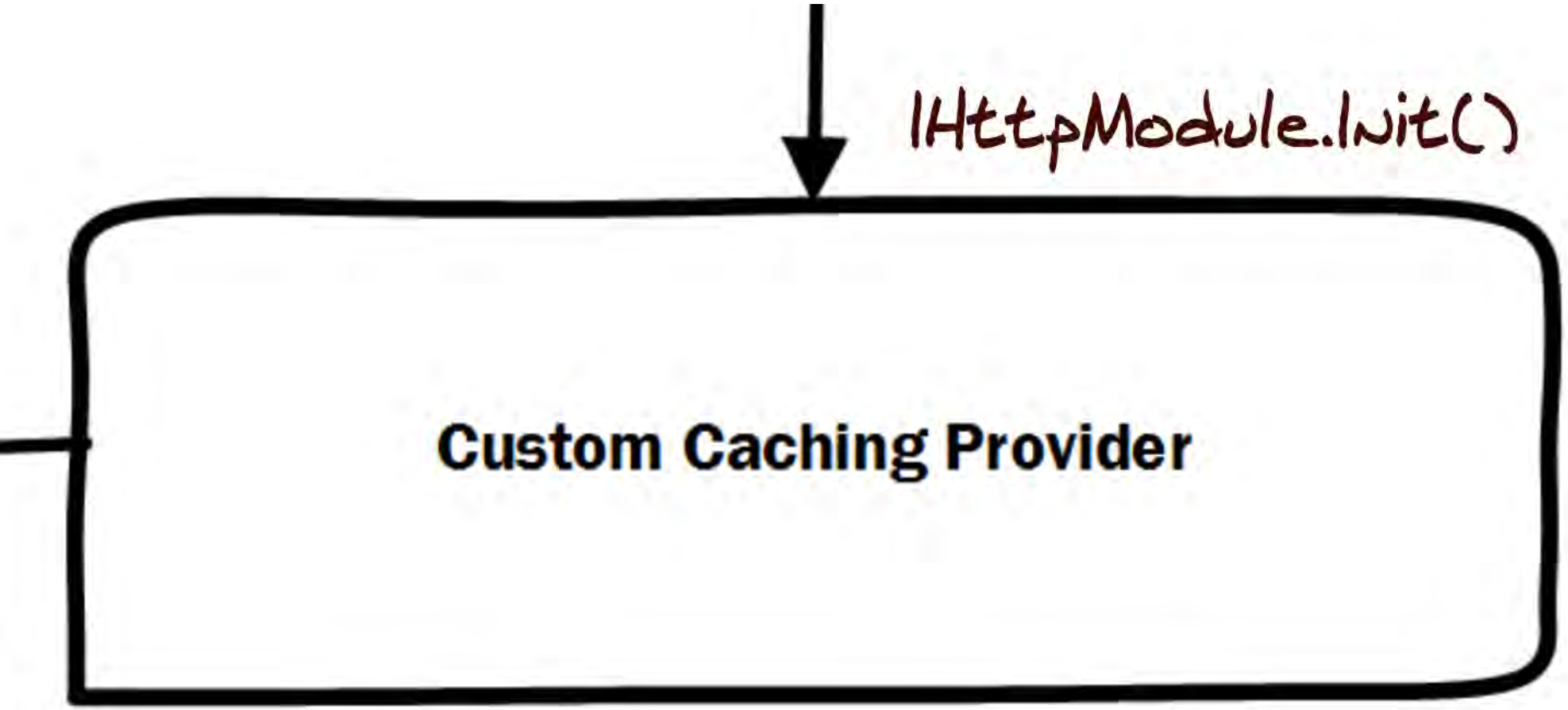
```
40
                0 references
41              public override void FeatureActivated(SPFeatureReceiverProperties properties)
42              {
43                  // We need to do registration here to ensure that HttpModules are wired
44                  // into the web.config. Grab a few references and needed names.
45                  SPWebApplication targetWebApp = ((SPSite)properties.Feature.Parent).WebApplication;
46                  String fullAssemblyName = Assembly.GetExecutingAssembly().FullName;
47                  String cachingClassName = typeof(HadACookieCustomModule).AssemblyQualifiedName;
48
49                  // Modification to register custom caching HttpModule
50                  SPWebConfigModification cachingHttpMod = new SPWebConfigModification();
51                  cachingHttpMod.Path = HTTP_MODULE_PATH;
52                  cachingHttpMod.Name = CACHING_HTTP_MODULE_NAME;
53                  cachingHttpMod.Sequence = 0;
54                  cachingHttpMod.Owner = FEATURE_OWNER;
55                  cachingHttpMod.Type = SPWebConfigModification.SPWebConfigModificationType.EnsureChildNode;
56                  cachingHttpMod.Value = String.Format(CACHING_HTTP_MODULE_VALUE, cachingClassName);
57
58                  // Apply the modifications and update the web.config file(s).
59                  targetWebApp.WebConfigModifications.Add(cachingHttpMod);
60                  targetWebApp.Update();
61                  targetWebApp.WebService.ApplyWebConfigModifications();
62              }
63
```

A: The caching module (an HttpModule) is wired into the ASP.NET pipeline using SPWebConfigModification entries in the FeatureActivated event of a FeatureReceiver

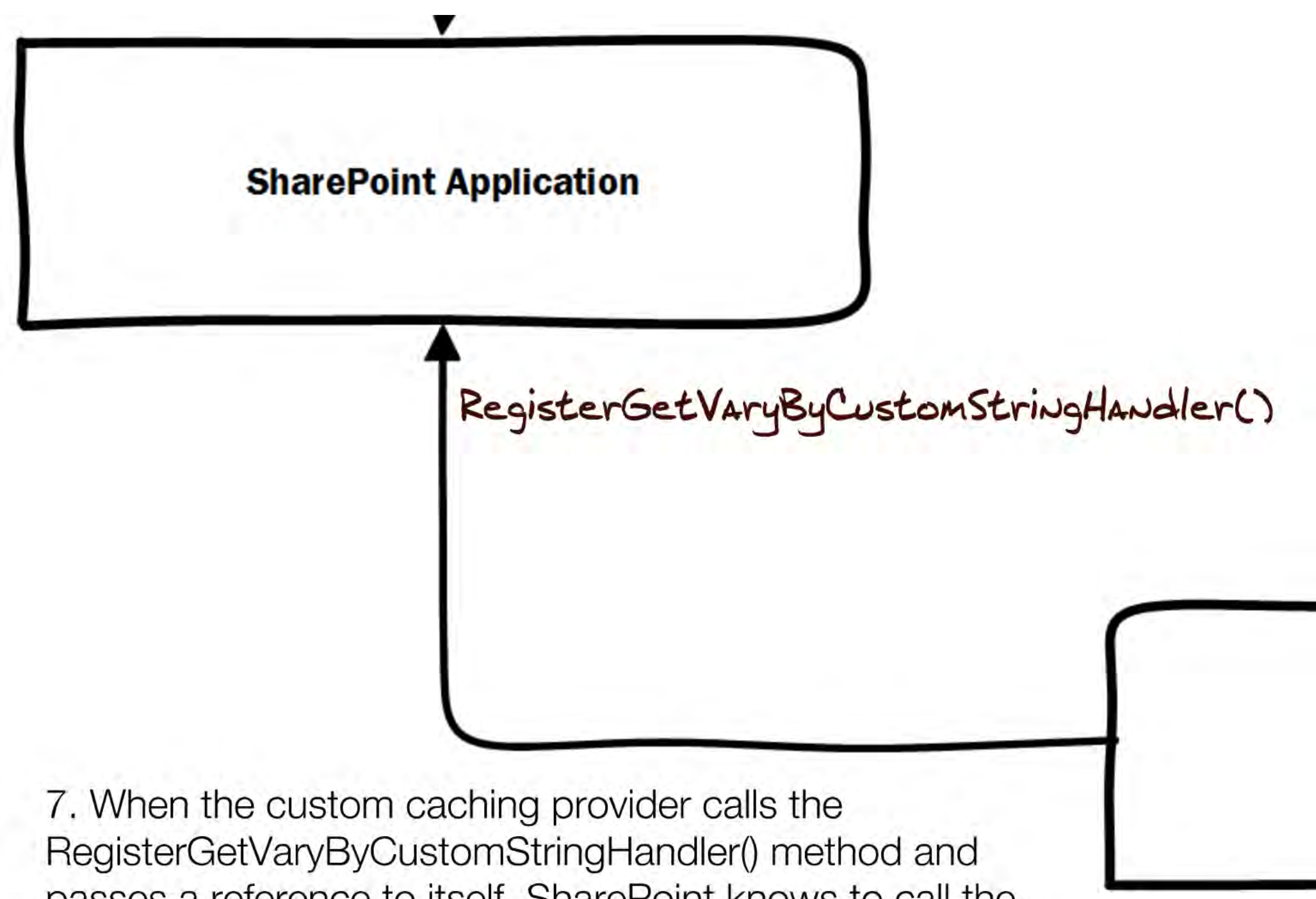
```
    </requestFiltering>
  </security>
  <validation validateIntegratedModeConfiguration="false" />
  <modules runAllManagedModulesForAllRequests="true">
    <remove name="AnonymousIdentification" />
    <remove name="FileAuthorization" />
    <remove name="Profile" />
    <remove name="WebDAVModule" />
    <remove name="Session" />
    <add name="SPNativeRequestModule" preCondition="integratedMode" />
    <add name="SPRequestModule" preCondition="integratedMode" type="Microsoft.SharePoint.ApplicationRuntime.SPRe
    <add name="ScriptModule" preCondition="integratedMode" type="System.Web.Handlers.ScriptModule, System.Web.Ex
    <add name="SharePoint14Module" preCondition="integratedMode" />
    <add name="StateServiceModule" type="Microsoft.Office.Server.Administration.StateModule, Microsoft.Office.Se
    <add name="PublishingHttpModule" type="Microsoft.SharePoint.Publishing.PublishingHttpModule, Microsoft.Share
    <add name="DesignHttpModule" preCondition="integratedMode" type="Microsoft.SharePoint.Publishing.Design.Desi
    <add name="FederatedAuthentication" type="Microsoft.SharePoint.IdentityModel.SPFederationAuthenticationModul
    <add name="SessionAuthentication" type="Microsoft.SharePoint.IdentityModel.SPSessionAuthenticationModule, Mi
    <add name="SPWindowsClaimsAuthentication" type="Microsoft.SharePoint.IdentityModel.SPWindowsClaimsAuthentica
    <add name="SPApplicationAuthentication" type="Microsoft.SharePoint.IdentityModel.SPApplicationAuthenticatior
    <add name="HadACookieHttpModule" type="SPMcDonough.CachingCodeSolutions.CcsExamples.HadACookieCustomModule,
  </modules>
  <handlers>
    <remove name="OPTIONSVerbHandler" />
    <remove name="WebServiceHandlerFactory-Integrated" />
    <remove name="WebDAV" />
    <add name="OwssvrHandler" scriptProcessor="C:\Program Files\Common Files\Microsoft Shared\Web Server Extensi
    <add name="ScriptHandlerFactory" verb="*" path="*.asmx" preCondition="integratedMode" type="System.Web.Scrip
    <add name="ScriptHandlerFactoryAppServices" verb="*" path="*_AppService.axd" preCondition="integratedMode" t
    <add name="ScriptResource" preCondition="integratedMode" verb="GET,HEAD" path="ScriptResource.axd" type="Sys
    <add name="ChartImg" verb="*" path="ChartImg.axd" type="System.Web.UI.DataVisualization.Charting.ChartHttpHa
    <add name="JSONHandlerFactory" path="*.json" verb="*" type="System.Web.Script.Services.ScriptHandlerFactory,
    <add name="CrossDomainAjaxOptions" verb="OPTIONS" path="CrossDomainAjax.ashx" resourceType="Unspecified" pre
    <add name="ReportViewerWebControl" verb="*" path="Reserved.ReportViewerWebControl.axd" type="Microsoft.Repor
    <remove name="ExtensionlessUrl-ISAPI-4.0_64bit" />
```

A section from the web.config

```
add name="PublishingHttpModule" type="
add name="DesignHttpModule" preConditi
add name="FederatedAuthentication" typ
add name="SessionAuthentication" type=
add name="SPWindowsClaimsAuthenticatio
add name="SPApplicationAuthentication"
add name="HadACookieHttpModule" type="
odules>
ndlers>
remove name="OPTIONSVerbHandler" />
remove name="WebServiceHandlerFactory-
remove name="WebDAV" />
add name="OwssvrHandler" scriptProcess
```



6. In the Init() method, the custom caching provider gets a reference to the SharePoint application. It uses that reference to call into SharePoint to register itself up for subsequent caching-related calls.

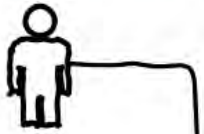


7. When the custom caching provider calls the RegisterGetVaryByCustomStringHandler() method and passes a reference to itself, SharePoint knows to call the provider anytime a page output request is being handled.

6. In the Init()

Application Setup

Assumption: application pool isn't spun-up yet.



1. A user requests a page from SharePoint (e.g., default.aspx)

5. Our custom caching provider is instantiated, and a reference to the SharePoint SPHttpApplication is passed in.

Q: How does SharePoint know about the custom caching module?

A: The caching module (an HttpModule) is wired into the ASP.NET pipeline using SPWebConfigModification entries in the FeatureActivated event of a FeatureReceiver

A section from the web.config

instantiate custom handler

ASP.NET Worker Process

instantiate SharePoint application

2. ASP.NET spins-up a worker process and an application pool (as needed) for SharePoint.

3. The web.config the entities it references (e.g., HttpHandlers and HttpModules) are processed for initialization.

4. An instance of the SharePoint application (SPHttpApplication) is started-up to process SharePoint requests (for example, our page request)

SharePoint Application

RegisterGetVaryByCustomStringHandler()

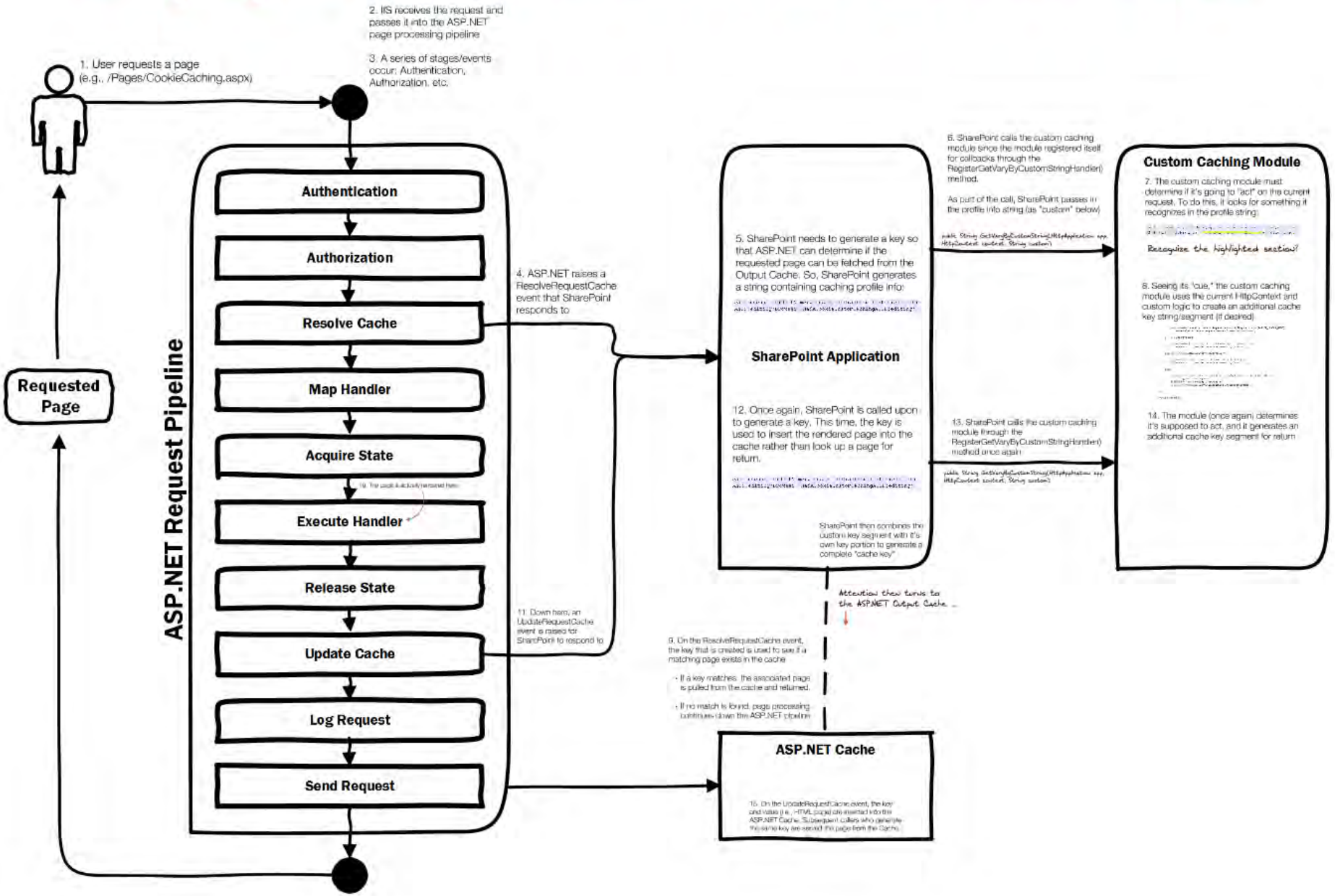
IHttpModule.Init()

Custom Caching Provider

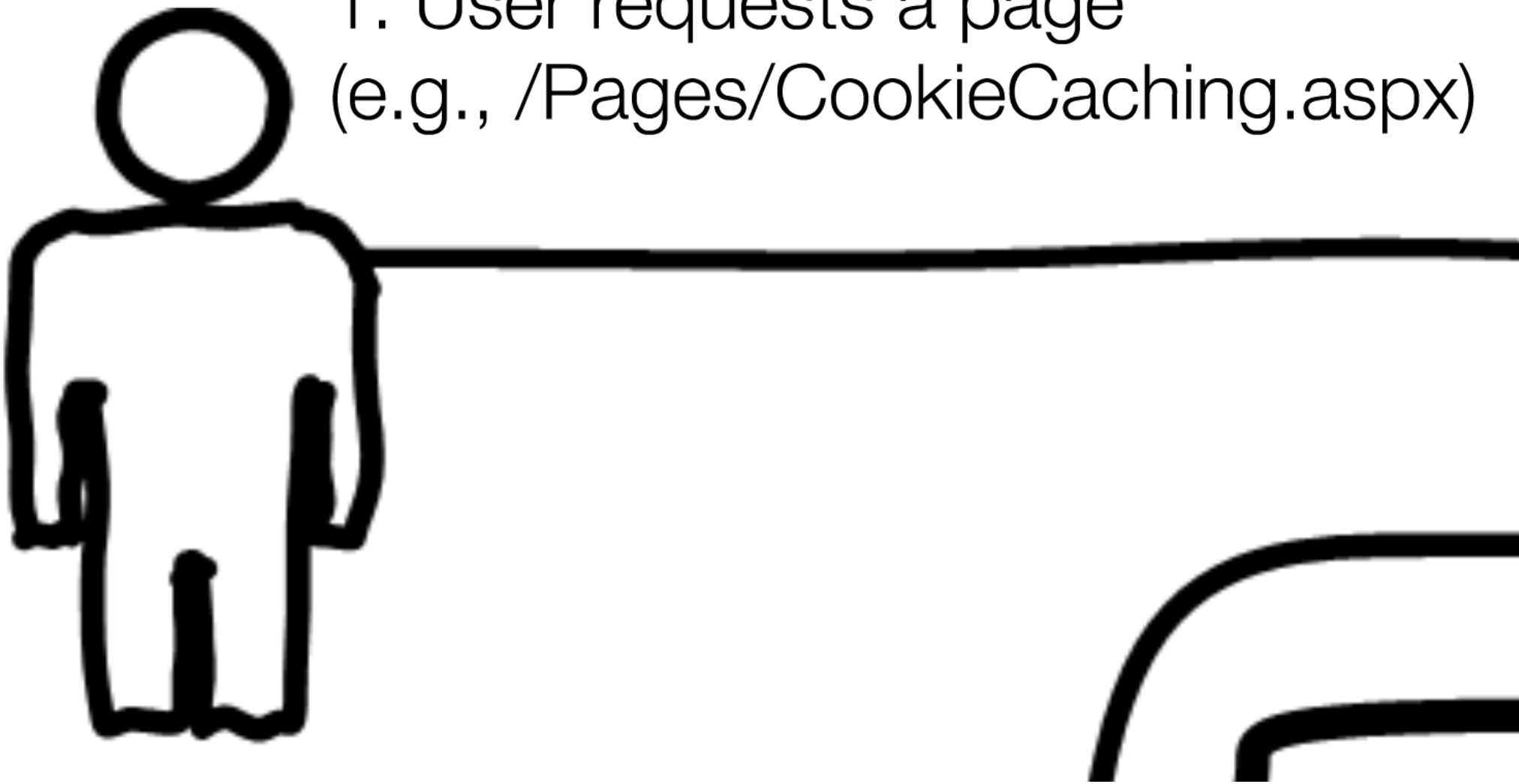
7. When the custom caching provider calls the RegisterGetVaryByCustomStringHandler() method and passes a reference to itself, SharePoint knows to call the provider anytime a page output request is being handled.

6. In the Init() method, the custom caching provider gets a reference to the SharePoint application. It uses that reference to call into SharePoint to register

Application Runtime



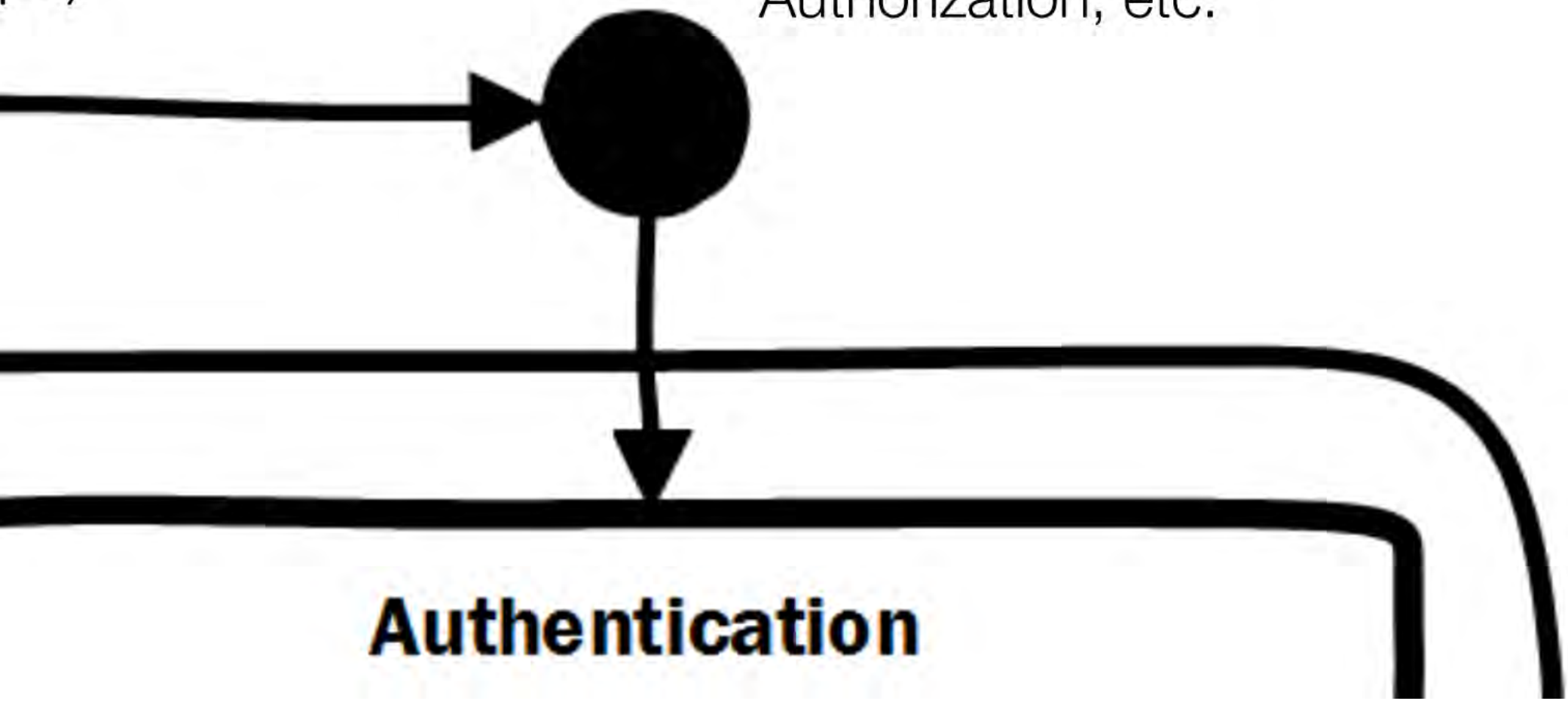
1. User requests a page
(e.g., /Pages/CookieCaching.aspx)

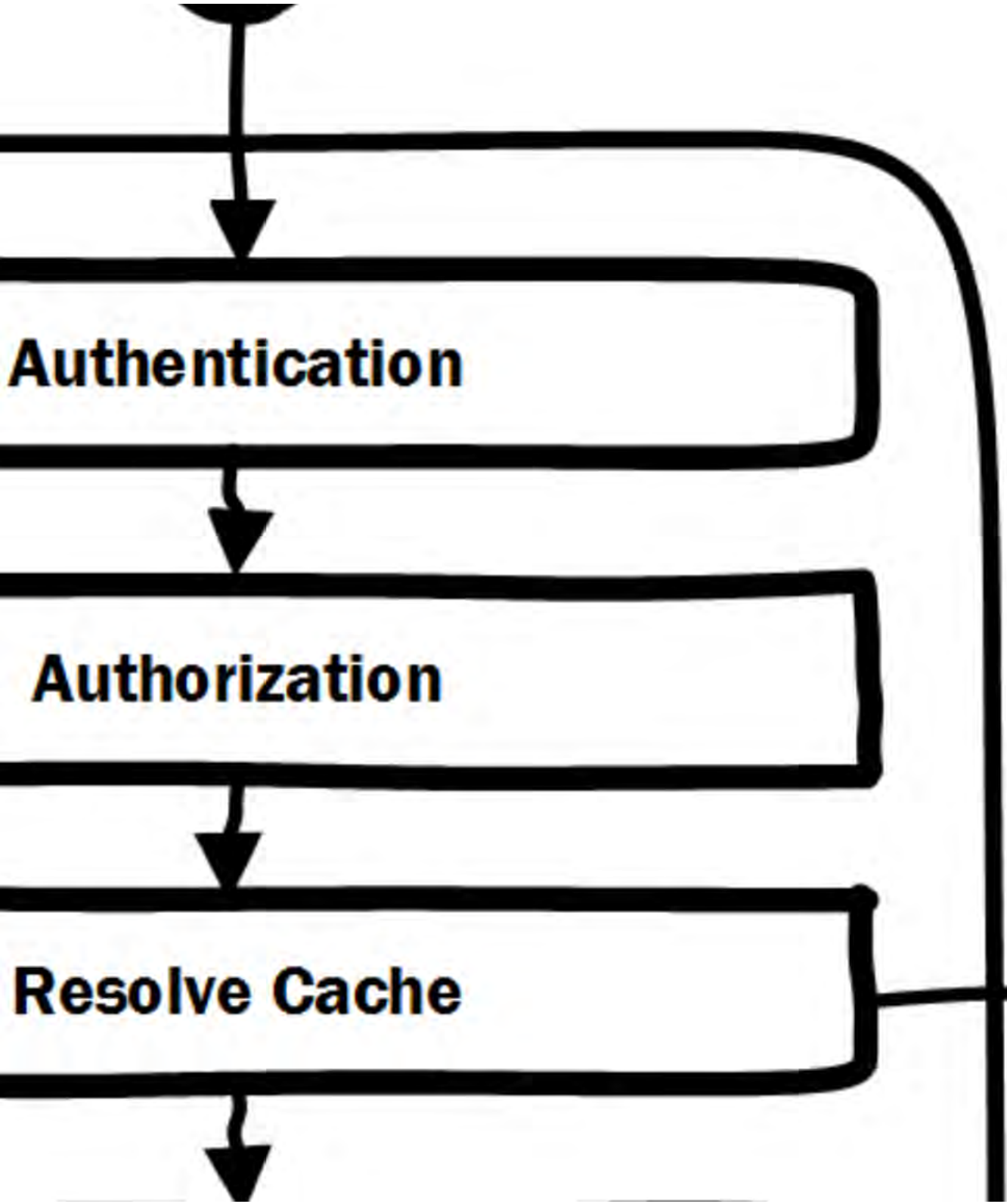


2. IIS receives the request and passes it into the ASP.NET page processing pipeline

3. A series of stages/events occur: Authentication, Authorization, etc.

px)





4. ASP.NET raises a ResolveRequestCache event that SharePoint responds to

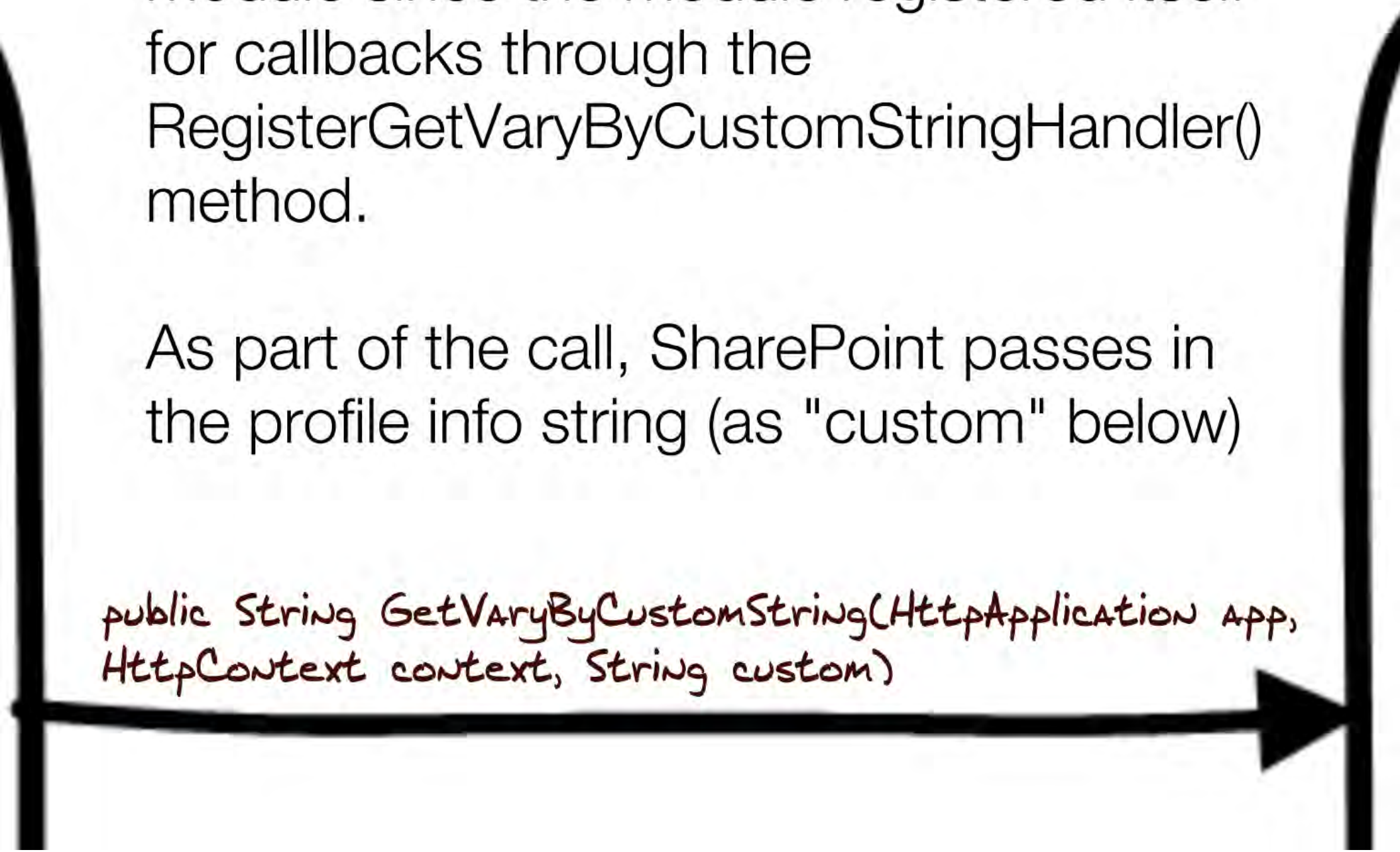
5. SharePoint needs to generate a key so that ASP.NET can determine if the requested page can be fetched from the Output Cache. So, SharePoint generates a string containing caching profile info:

```
cachingenabled;HostName;wpcustomized;authenticated;console;
ANON:editing;Browser, HadACookieCustomCachingAUTH:editing;
```

SharePoint Application

6. SharePoint calls the custom caching module since the module registered itself for callbacks through the RegisterGetVaryByCustomStringHandler() method.

As part of the call, SharePoint passes in the profile info string (as "custom" below)



Custom Caching Module

7. The custom caching module must determine if it's going to "act" on the current request. To do this, it looks for something it recognizes in the profile string:

```
cachingenabled;HostName;wpcustomized;authenticated;console;
ANON:editing;Browser, HadACookieCustomCachingAUTH:editing;
```

Recognize the highlighted section?

8. Seeing its "cue," the custom caching module uses the current HttpContext and custom logic to create an additional cache key string/segment (if desired)

```
    Boolean isHeaderPresent = context.Request.Headers.AllKeys.Contains(TARGET_HEADER_NAME);
    String headerValue = context.Request.Headers[TARGET_HEADER_NAME];

    if (!isHeaderPresent)
    {
        // No header is present; return a cache key for general use
        cacheKey = String.Format(CACHE_KEY_TEMPLATE, "ASBSENT");
    }
    else if (String.IsNullOrEmpty(headerValue))
    {
        // Header is present but no per-user value is assigned.
        cacheKey = String.Format(CACHE_KEY_TEMPLATE, "PRESENT");
    }
    else
    {
        // Header is present and a (potentially) unique value is assigned. Disable
        // caching for this request.
        cacheKey = Guid.NewGuid().ToString();
        PublishingHttpModule.DontEnableCachingForRequest(context);
    }
}

return cacheKey;
```

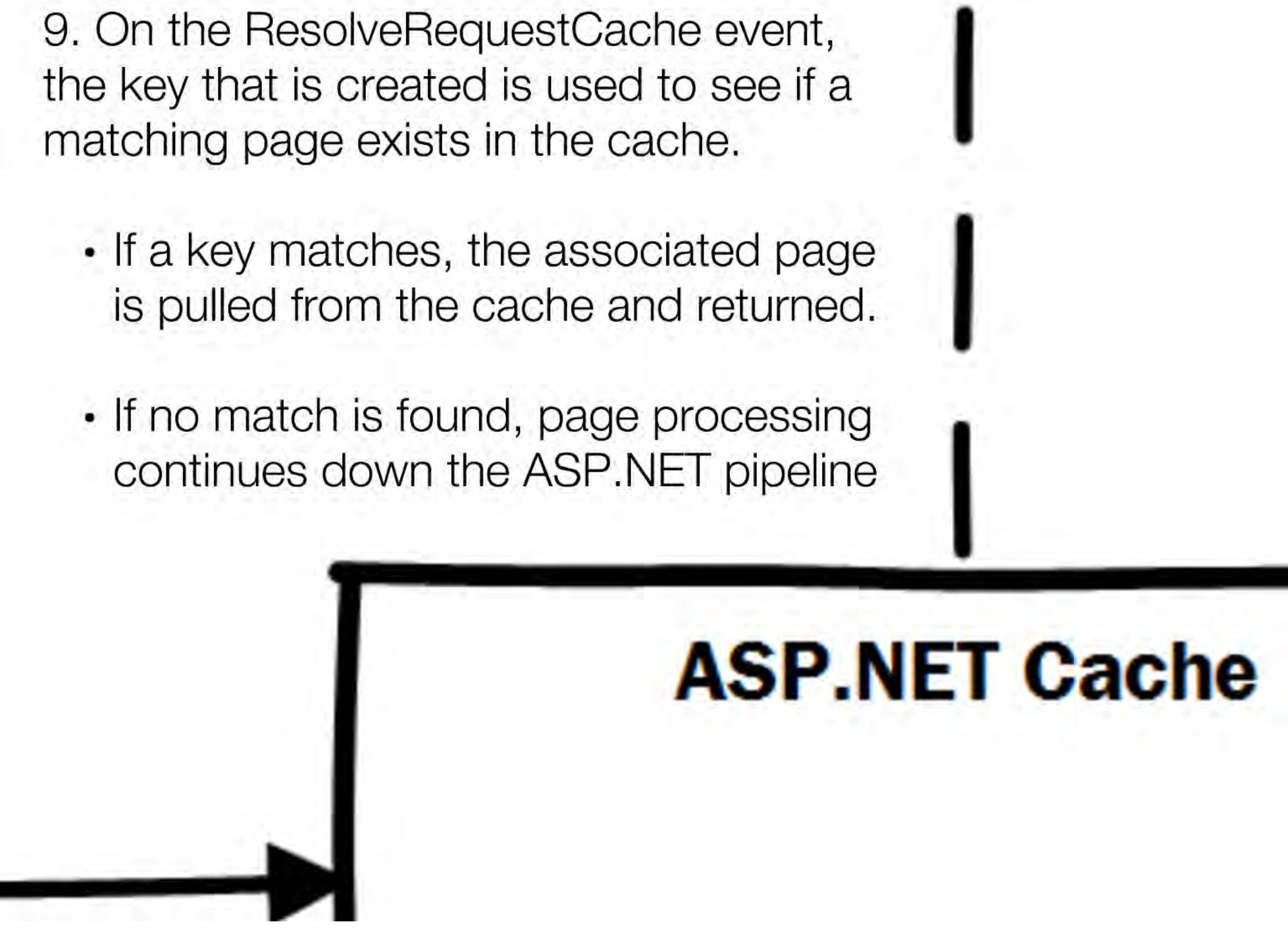
```
e if (String.IsNullOrEmpty(headerValue))

 // Header is present but no per-user value is assigned.
 cacheKey = String.Format(CACHE_KEY_TEMPLATE, "PRESENT");

e

 // Header is present and a (potentially) unique value is as
 // caching for this request.
 cacheKey = Guid.NewGuid().ToString();
 PublishingHttpModule.DontEnableCachingForRequest(context);



cacheKey;
```

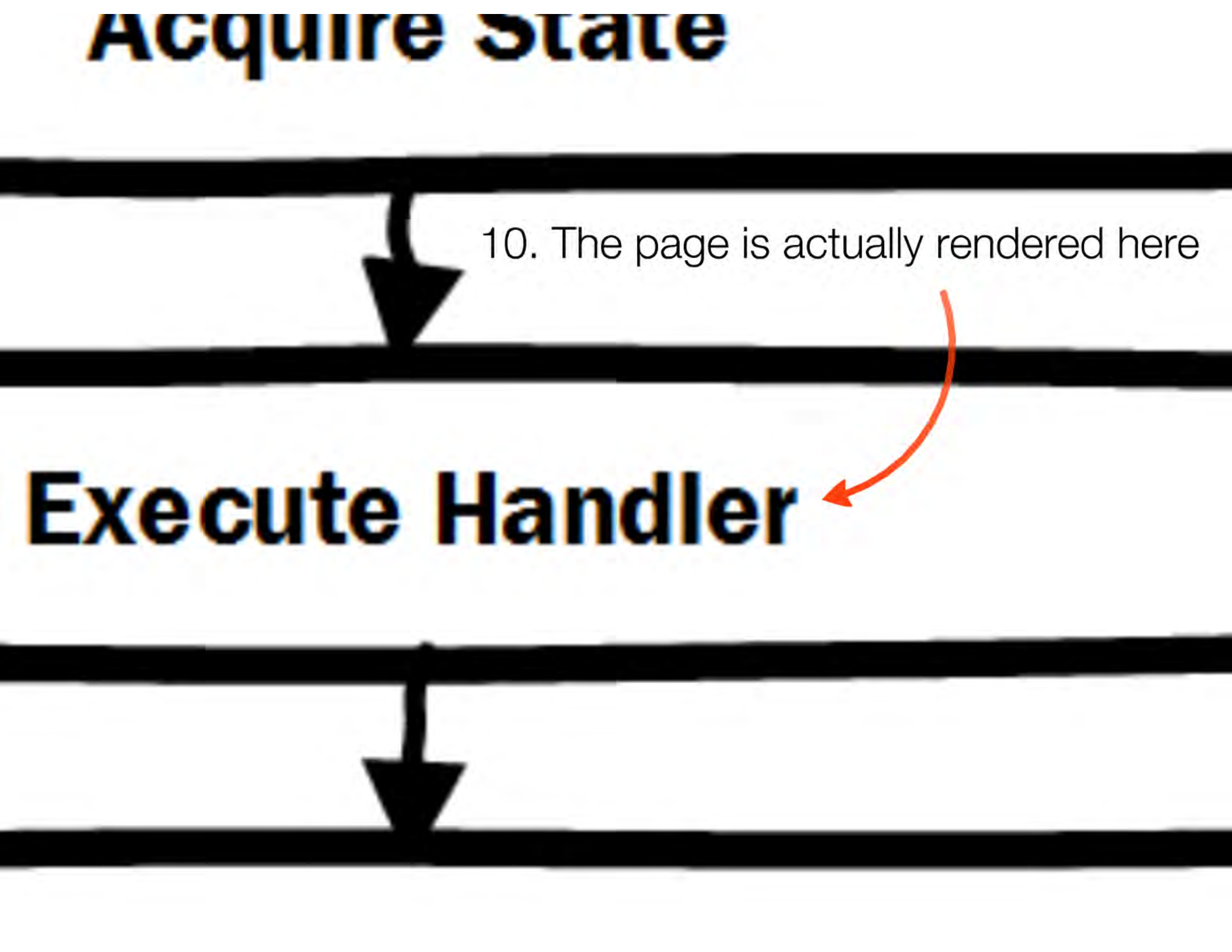
SharePoint then combines the custom key segment with it's own key portion to generate a complete "cache key"

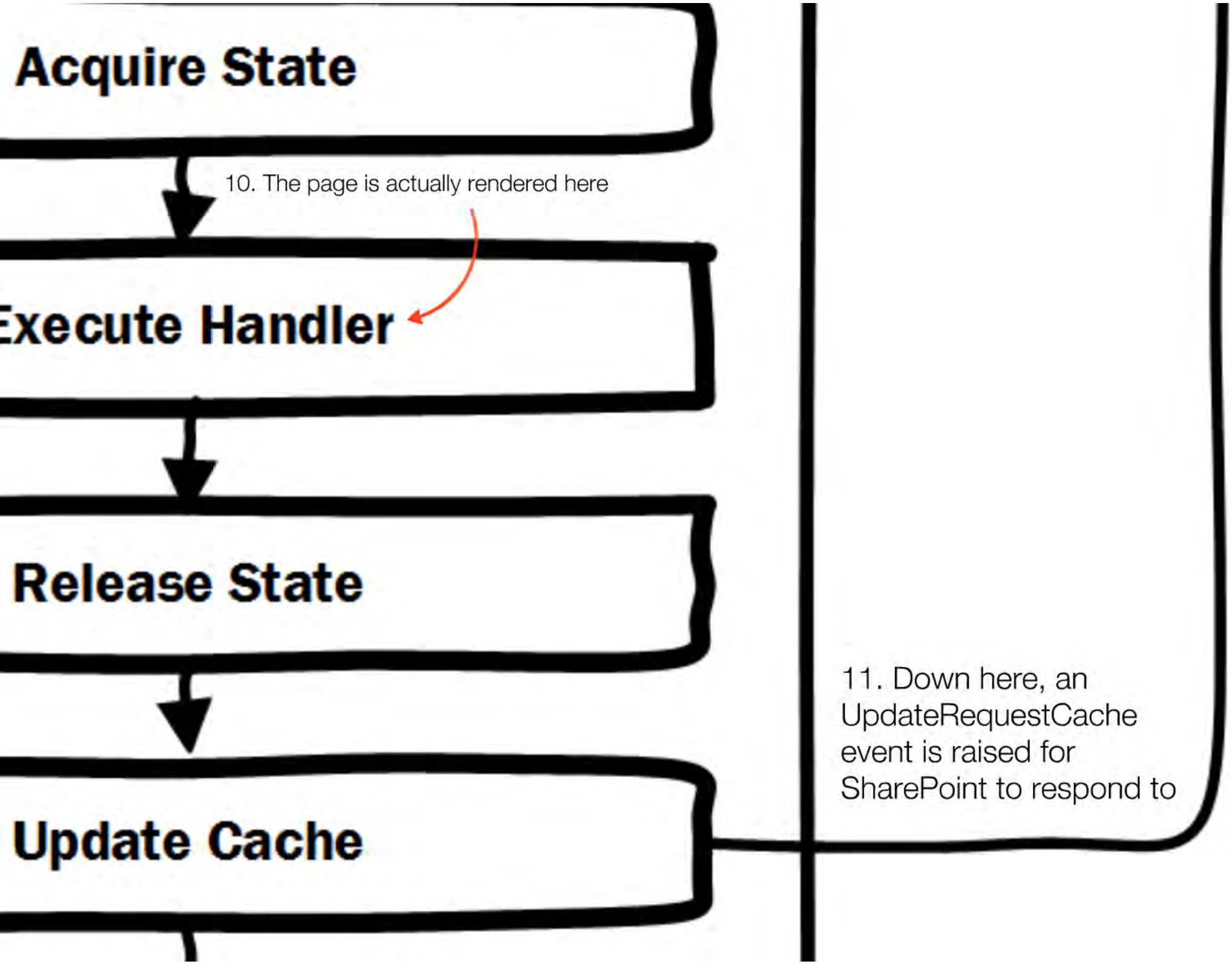
Attention then turns to the ASP.NET Output Cache ...

9. On the ResolveRequestCache event, the key that is created is used to see if a matching page exists in the cache.

- If a key matches, the associated page is pulled from the cache and returned.
- If no match is found, page processing continues down the ASP.NET pipeline




Acquire State
10. The page is actually rendered here
Execute Handler

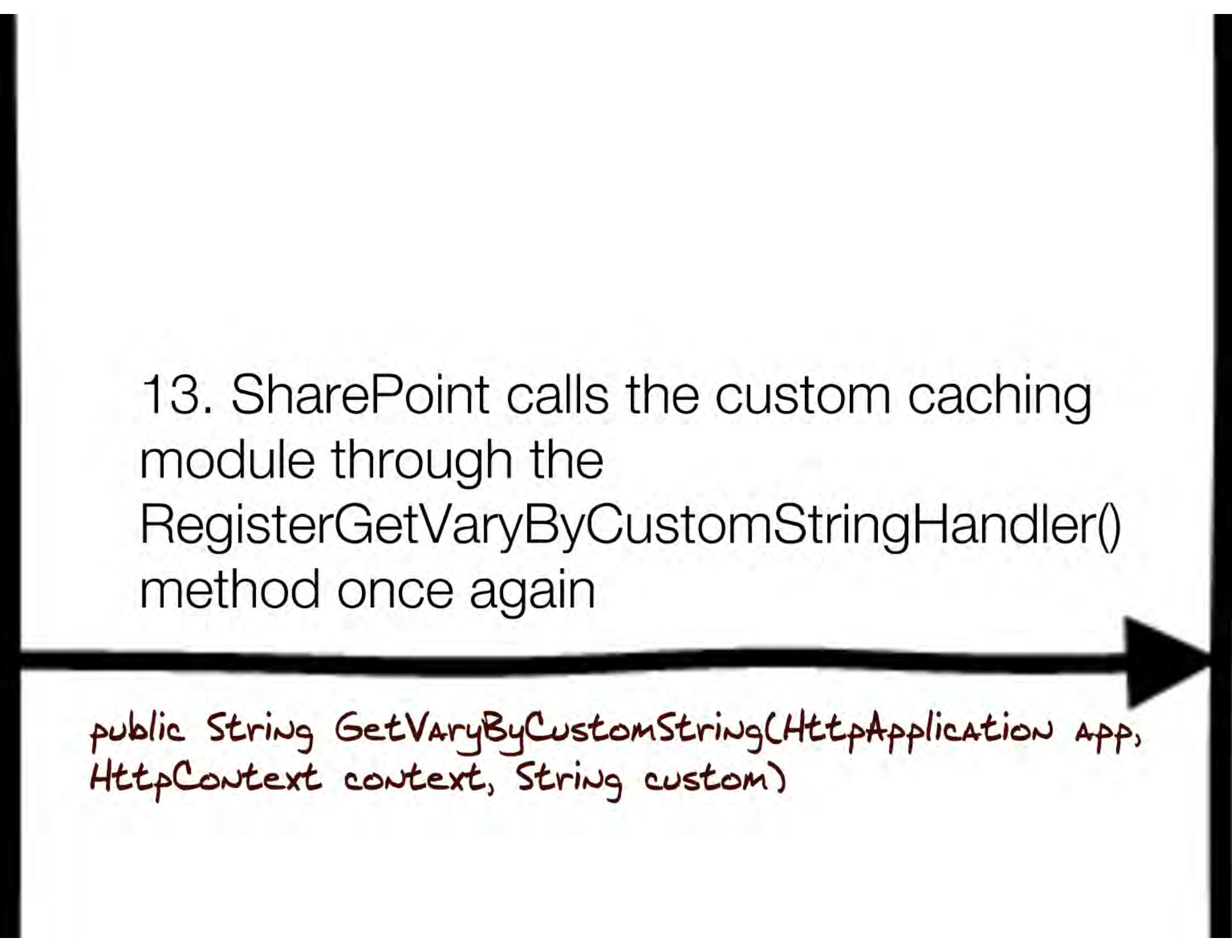

Acquire State
10. The page is actually rendered here
Execute Handler
Release State
Update Cache
11. Down here, an UpdateRequestCache event is raised for SharePoint to respond to

SharePoint Application

12. Once again, SharePoint is called upon to generate a key. This time, the key is used to insert the rendered page into the cache rather than look up a page for return.

```
cachingenabled;HostName;wpcustomized;authenticated;console;
ANON:editing;Browser, HadACookieCustomCachingAUTH:editing;
```

13. SharePoint calls the custom caching module through the RegisterGetVaryByCustomStringHandler() method once again



```
public String GetVaryByCustomString(HttpApplication app,
HttpContext context, String custom)
```

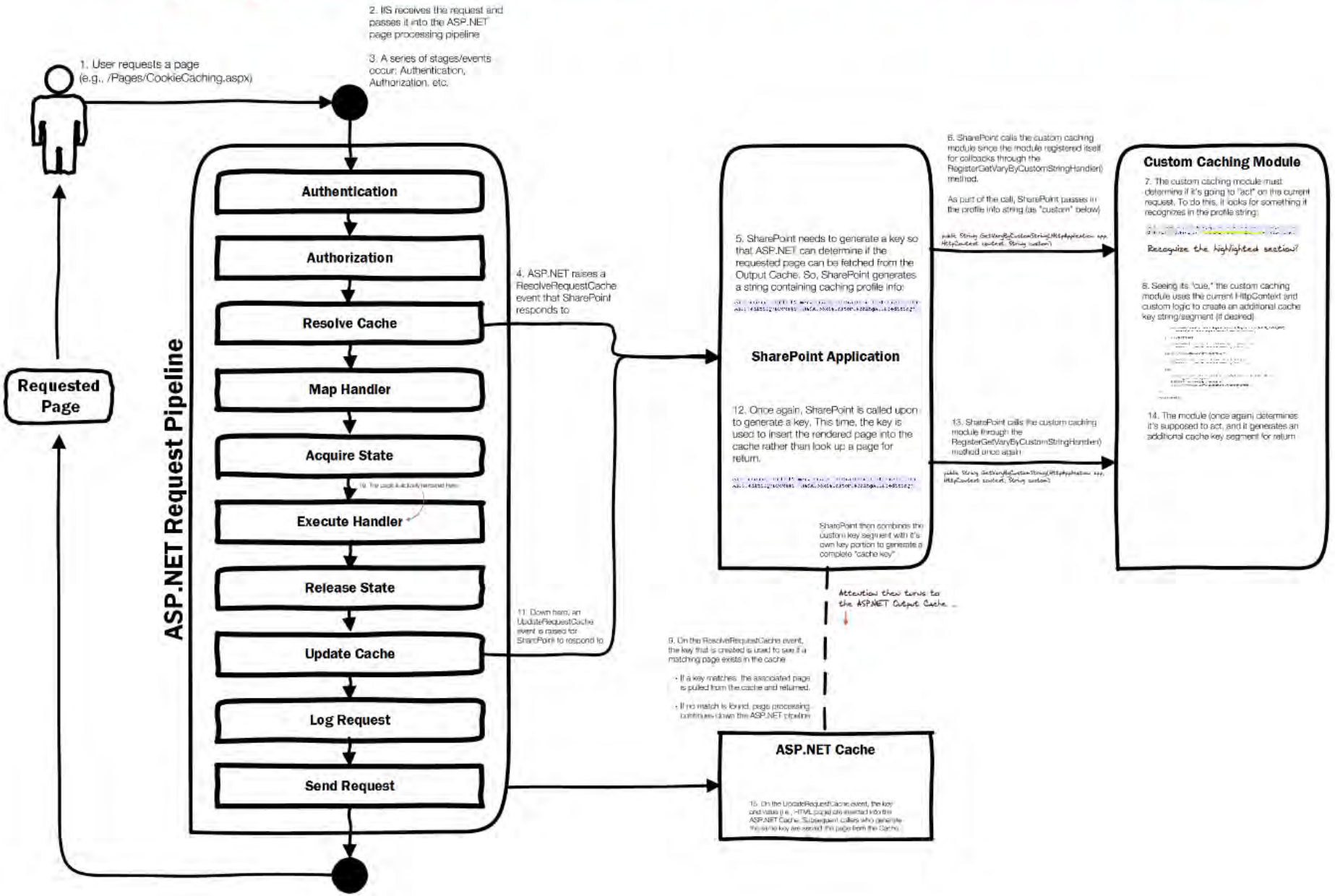
```
    Boolean isHeaderPresent = context.Request.Headers.AllKeys.Contains(TARGET_HEADER_NAME);
    String headerValue = context.Request.Headers[TARGET_HEADER_NAME];

    if (!isHeaderPresent)
    {
        // No header is present; return a cache key for general use
        cacheKey = String.Format(CACHE_KEY_TEMPLATE, "ASBSENT");
    }
    else if (String.IsNullOrEmpty(headerValue))
    {
        // Header is present but no per-user value is assigned.
        cacheKey = String.Format(CACHE_KEY_TEMPLATE, "PRESENT");
    }
    else
    {
        // Header is present and a (potentially) unique value is assigned. Disable
        // caching for this request.
        cacheKey = Guid.NewGuid().ToString();
        PublishingHttpModule.DontEnableCachingForRequest(context);
    }
}

return cacheKey;
```

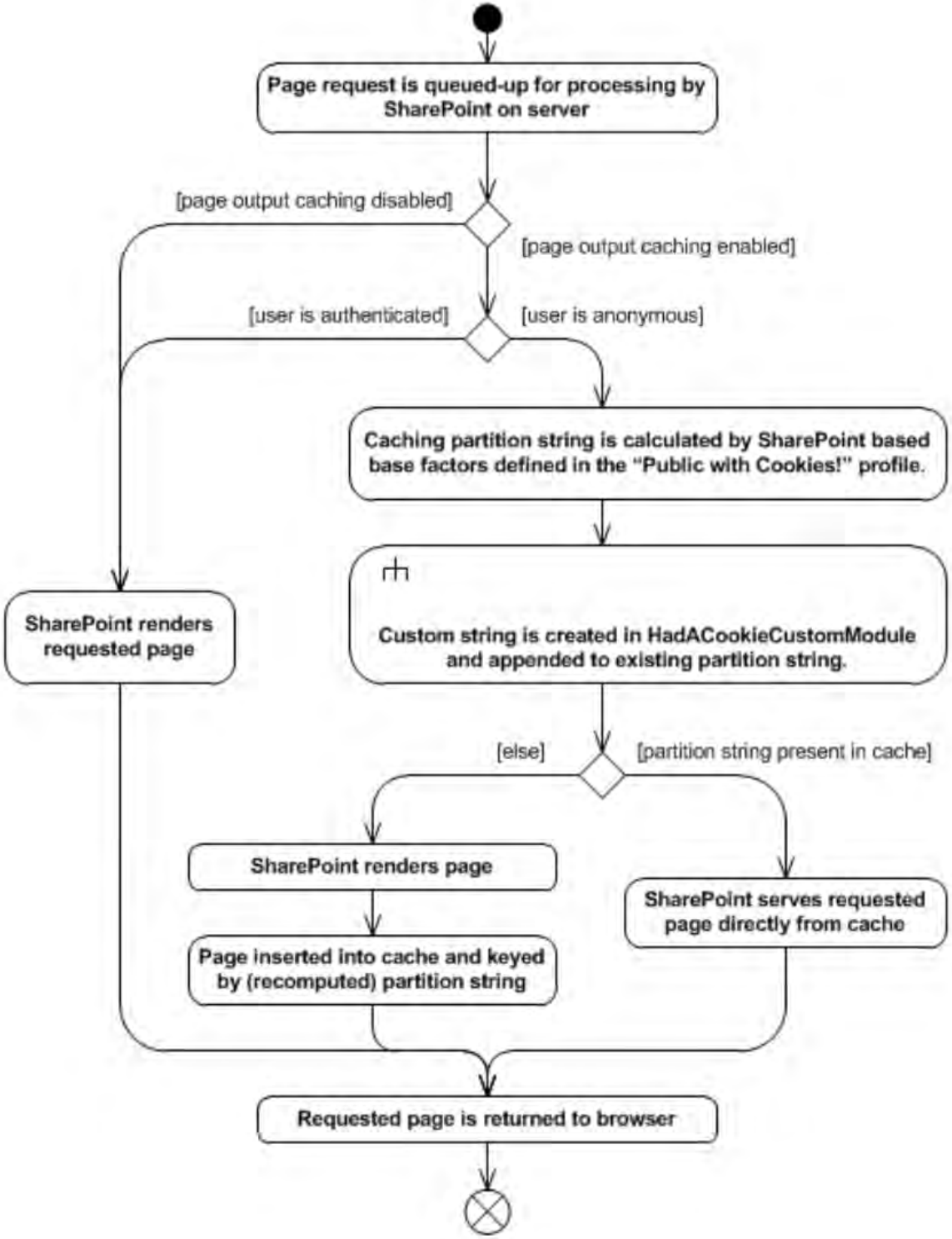
14. The module (once again) determines it's supposed to act, and it generates an additional cache key segment for return

ASP.NET Cache

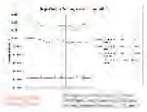
15. On the UpdateRequestCache event, the key and value (i.e., HTML page) are inserted into the ASP.NET Cache. Subsequent callers who generate the same key are served the page from the Cache.

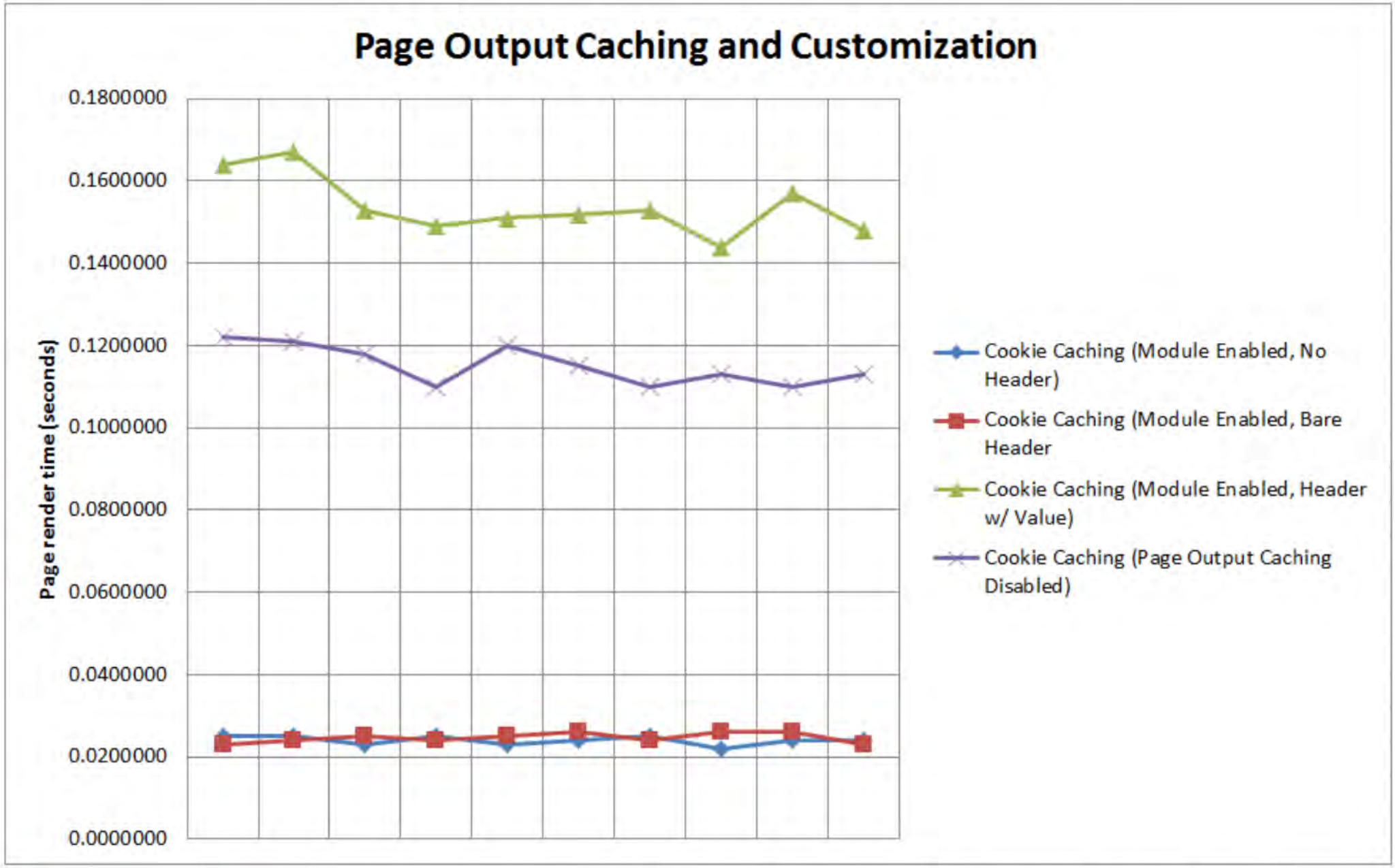
Application Runtime




Page request is queued-up for processing by SharePoint on server
[page output caching disabled]
[page output caching enabled]
[user is authenticated]
[user is anonymous]
Caching partition string is calculated by SharePoint based base factors defined in the "Public with Cookies!" profile.
SharePoint renders requested page
Custom string is created in HadACookieCustomModule and appended to existing partition string.
[else]
[partition string present in cache]
SharePoint renders page
SharePoint serves requested page directly from cache
Page inserted into cache and keyed by (recomputed) partition string
Requested page is returned to browser

Demo





Average Page Render Times

- No Page Output Caching: 0.1152 sec
- With Page Output Caching: 0.0243 sec
- Actively Disabling Caching: 0.1538 sec !!!!

Limitations and Watch-Outs

- Don't forget to include the "Vary by Custom Parameter" in your cache profile - and check for it in the GetVaryByCustomString method
- If your code isn't getting called, ensure the HttpModule is properly wired-up
- Remember that GetVaryByCustomString can be called twice in a single page request: once for lookup, and second for cache insertion*
- Avoid any costly or long-running operations in your GetVaryByCustomString method

Sum-up: The nuclear option. In my experience, this is a last resort - not the place to actually start

Second call (for insertion) only happens when page is being rendered - either on initial insert or re-rendering following ejection (cache time elapsed)

Limitations and Watch-Outs

- Don't forget to include the "Vary by Custom Parameter" in your cache profile - and check for it in the GetVaryByCustomString method
- If your code isn't getting called, ensure the HttpModule is properly wired-up
- Remember that GetVaryByCustomString can be called twice in a single page request: once for lookup, and second for cache insertion*
- Avoid any costly or long-running operations in your GetVaryByCustomString method

Sum-up: The nuclear option. In my experience, this is a last resort - not the place to actually start

References

Cache Class (System.Web.Caching)
http://msdn.microsoft.com/en-us/library/system.web.caching.cache.aspx

AppFabric 1.1 for Windows Server
http://msdn.microsoft.com/en-us/windowsserver/ee695849

Improve performance of your SharePoint 2010 applications using Windows Server AppFabric caching
http://www.wictorwilen.se/Post/Improve-performance-of-your-SharePoint-2010-applications-using-Windows-Server-AppFabric-caching.aspx

Plan for feeds and the Distributed Cache service in SharePoint Server 2013
http://technet.microsoft.com/en-us/library/jj219572.aspx

How To Perform Fragment Caching in ASP.NET by Using Visual C#.NET
http://support.microsoft.com/kb/308378

OutputCacheParameters Class
http://msdn.microsoft.com/en-us/library/ms153449(v=vs.90)

References

Pages, Parsing, and Safe Mode
http://msdn.microsoft.com/en-us/library/gg552610.aspx#BKMK_PagesUI

Dynamically Updating Portions of a Cached Page
http://msdn.microsoft.com/en-us/library/ms227429(v=vs.90).aspx

Substitution Class
http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.substitution(v=vs.90).aspx

How to: Extend Caching by Using the VaryByCustom Event Handler in SharePoint Server 2010 (ECM)
http://msdn.microsoft.com/en-us/library/ms550239.aspx

When Page Output Caching Does Not Output
http://todd-carter.com/post/2012/01/31/When-Page-Output-Caching-Does-Not-Output.aspx

Fiddler Web Debugger - Script Samples
http://www.fiddlertool.com/Fiddler/dev/ScriptSamples.asp

Html Agility Pack
http://htmlagilitypack.codeplex.com/

References

Cumulative update package 6 for Microsoft AppFabric 1.1 for Windows Server

http://support.microsoft.com/kb/3042099

Managing Security (Windows Server AppFabric Caching)

http://msdn.microsoft.com/en-us/library/ff921012(v=azure.10).aspx

Sean P. McDonough

SharePoint Gearhead,
Developer, and Problem-Solver

My Employer: cardinal
PEOPLE.IDEAS.SOLUTIONS.

Twitter: @spmcdonough

Blog: http://SharePointInterface.com

About: http://about.me/spmcdonough