Solution Cache: Remote Stores

Elton Stoneman
elton@sixeyed.com

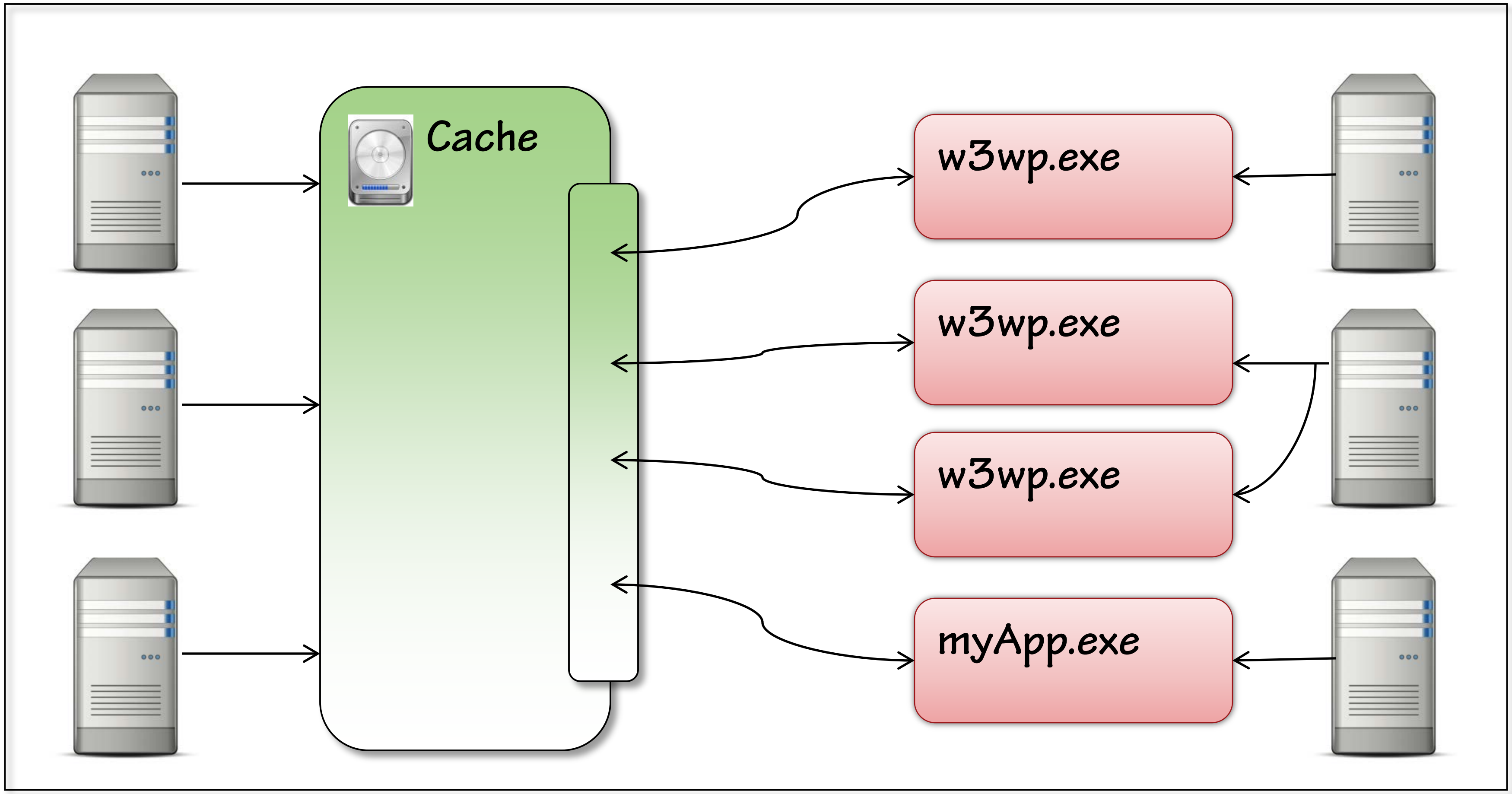
pluralsight
hardcore developer training

Outline

- **Remote cache stores**
  - Centralized
  - Distributed
  - Replicated
- **Remote caches**
  - Dependencies, usage and configuration
  - Management and extras
  - Matching against the decision matrix
- **Memcached**
- **Azure Table Storage**
- **DiskCache**
- **Applying the Decision Matrix**
  - Should I have multiple cache stores?

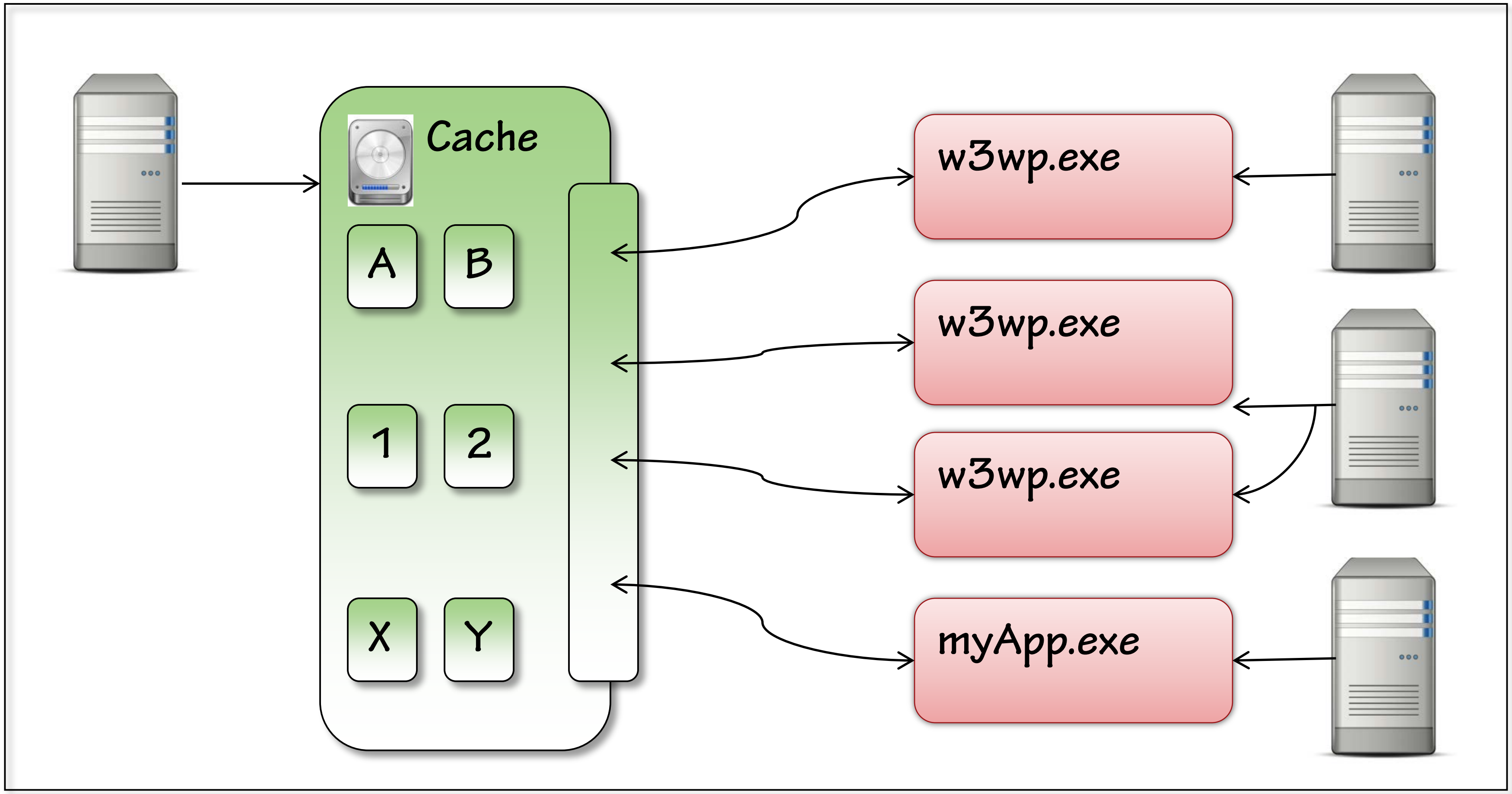
Remote Cache Stores

- Overview



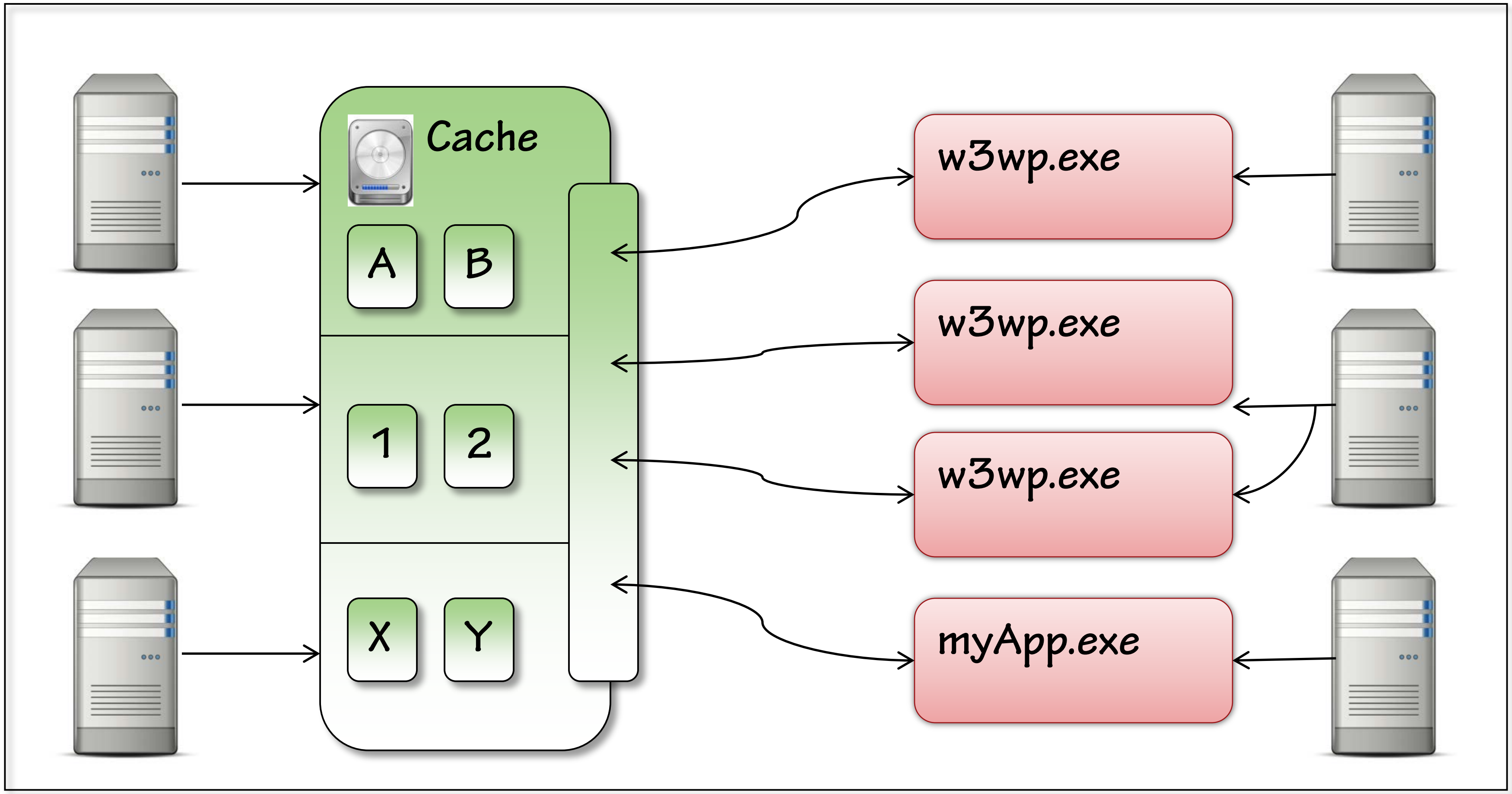
Remote Cache Stores

- Centralized



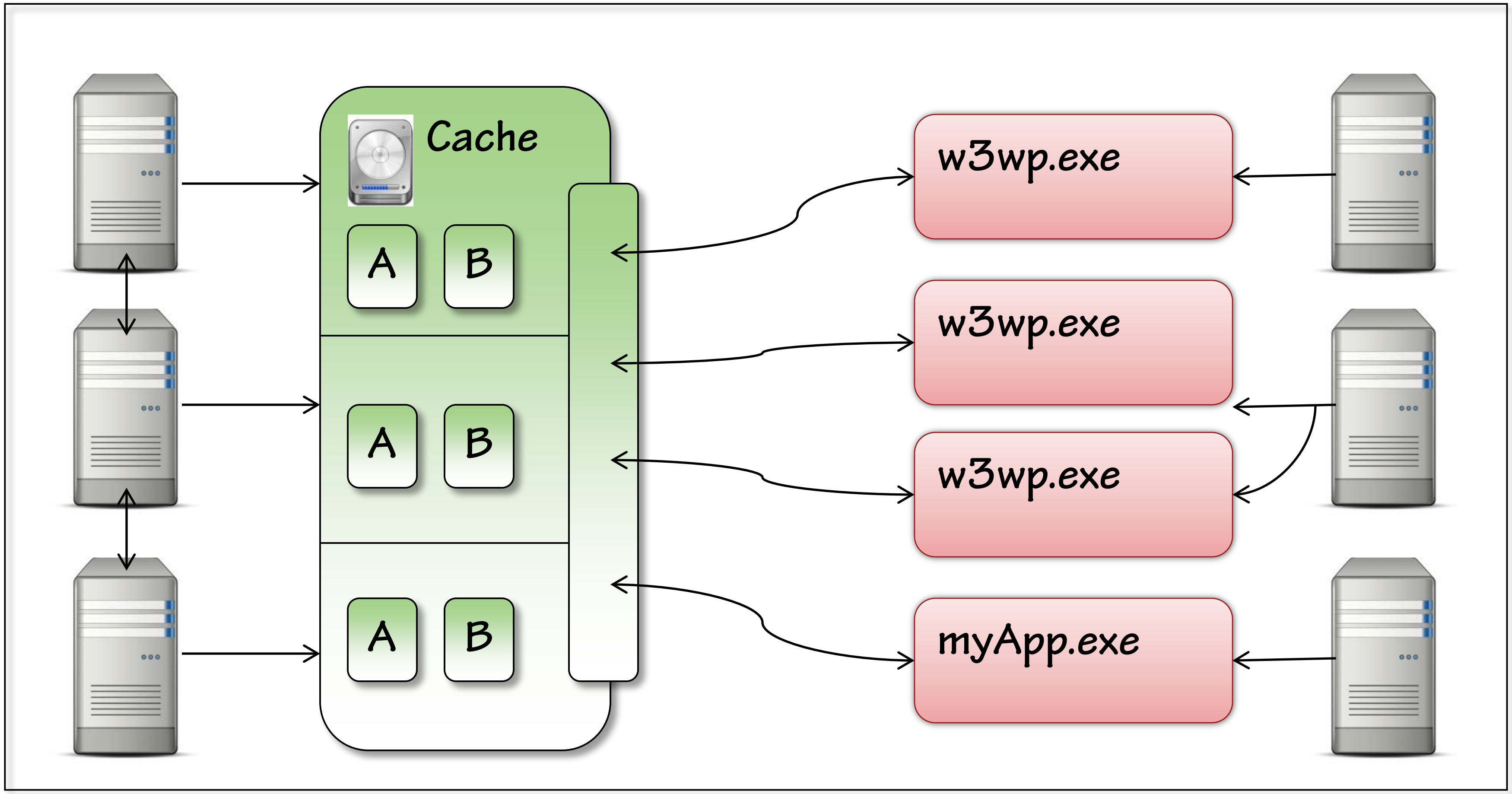
Remote Cache Stores

- Distributed



Remote Cache Stores

- Replicated



Memcached

- **Distributed cache**
- **Open source, cross-platform**
  - Evolved on Linux
  - Migrated to Windows with .NET client
  - Consistent protocol
- **Widely used**
  - Flickr, Twitter, Wikipedia, YouTube…

Memcached

- Demo

Memcached

- **Distributed cache**
  - Runs in console or Windows Service
  - Cache items split across nodes
- **Client configuration**

```xml
<enyim.com>
   <memcached protocol="Binary">
     <servers>
       <add address="127.0.0.1" port="11211" />
     </servers>
   </memcached>
```

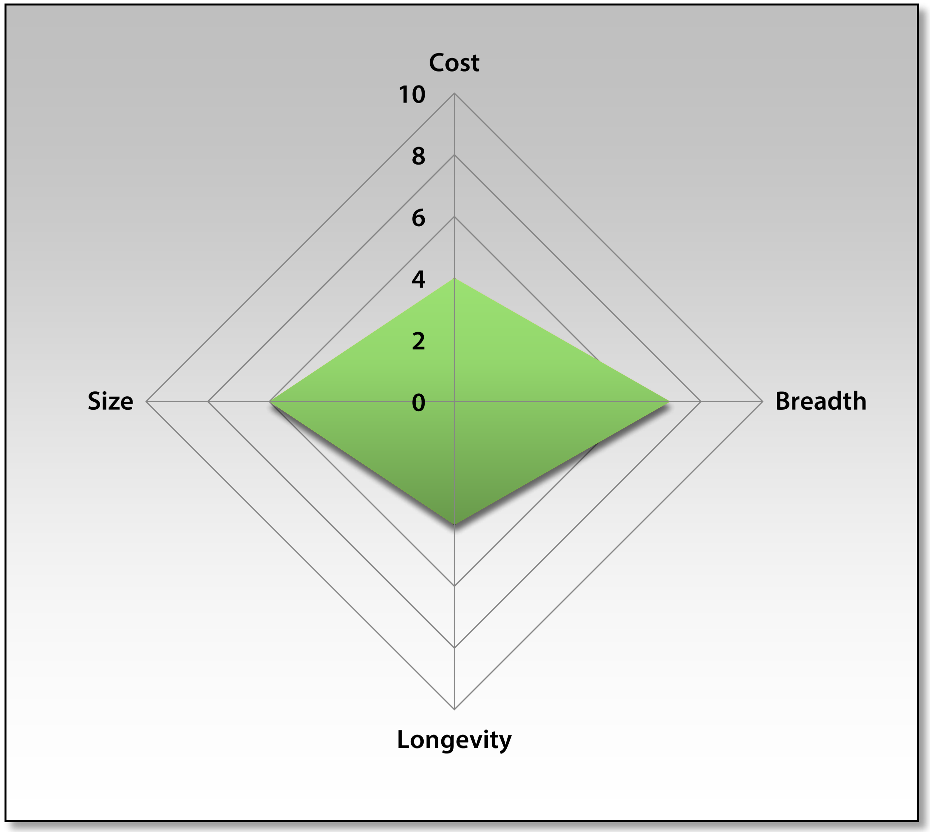
- **No out-of-the-box management interface**
  - Open protocol, 3rd party tools: Memcached Manager
- **Advanced features**
  - Socket pool; performance monitors
  - Protocol – AWS ElastiCache; EngineYard; Google App Engine

Memcached

- **Suitability applied to the decision matrix**
  - Cost – moderate, network access + memory lookup
  - Breadth – available to any process, any machine with network access
  - Longevity – machine uptime & network availability
  - Size
    - Total: sum of available memory on all nodes
    - Item: limited to 1Mb

Memcached

- Widely suitable, unless performance-critical



Azure Table Storage

- **Centralized cache**
  - PaaS ”no-SQL” db used as store
- **Custom**
  - Simple implementation
  - Supports ICache and expiration
- **Tried and tested**
  - Extensible

Azure Table Storage

- Demo

Azure Table Storage

- **Centralized cache**
  - PaaS storage, shared logical node
  - NuGet client library
- **Connection string configuration**

```
<connectionStrings>
   <!-- Azure Table Storage -->
   <add name="Sixeyed.Core.Cache"
        connectionString="DefaultEndpointsProtocol=https;
                          AccountName=sccache;
                          AccountKey=RW/LKHfeOBaYvjzIHZifXTsx…
```

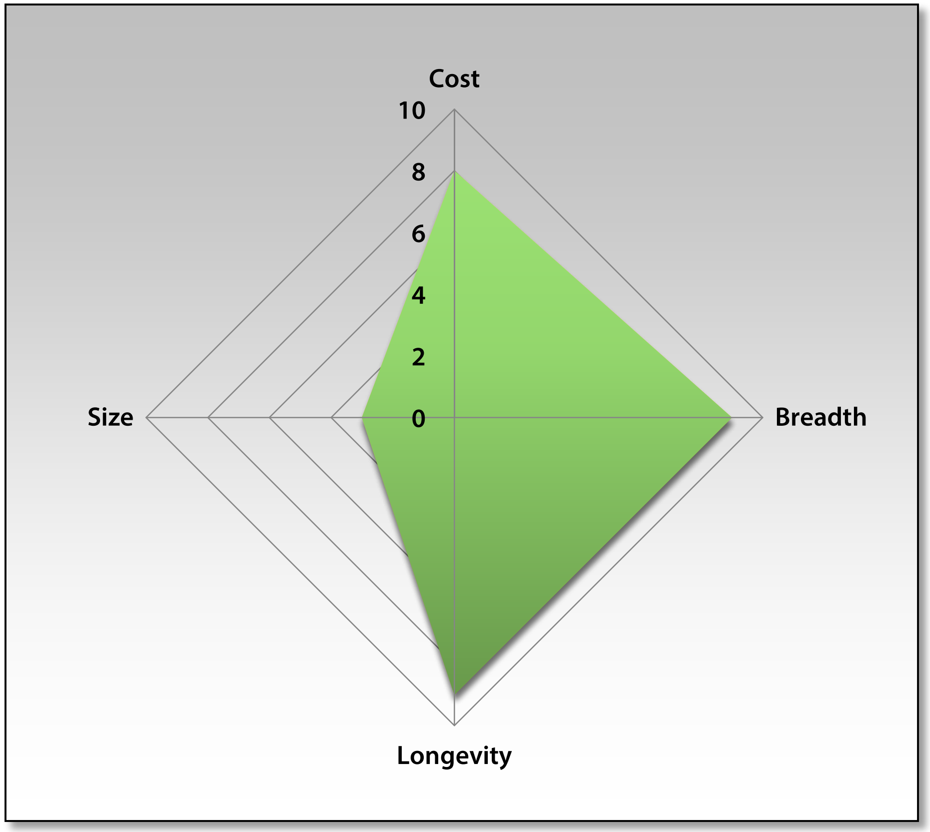
- **Management tools**
  - Azure Portal
  - Azure Storage Explorer
- **Extra features**
  - Persistence; geo-replication

Azure Table Storage

- **Suitability applied to the decision matrix**
  - Cost – slow, Internet access + disk IO
  - Breadth – available to any process, any machine with Internet access
  - Longevity – replication & network availability
  - Size
    - Total: Internet scale (100Tb)
    - Item: limited to 64Kb (1Mb with modification)

Azure Table Storage

- Good for smallish, long-lived items, balanced against performance



DiskCache

- **Centralized cache**
  - Items stored on network share
- **Custom**
  - Simple implementation
  - Supports ICache, cache size and expiration
- **Tried and tested**

DiskCache

- Demo

DiskCache

- **Centralized cache**
  - File system store
  - Single logical node
- **Simple configuration**

```
<sixeyed.core.caching>
   <diskCache path="\\192.168.2.160\cache"
              encryptItems="false"
              maxSizeInMb="200"/>
 </sixeyed.core.caching>
```

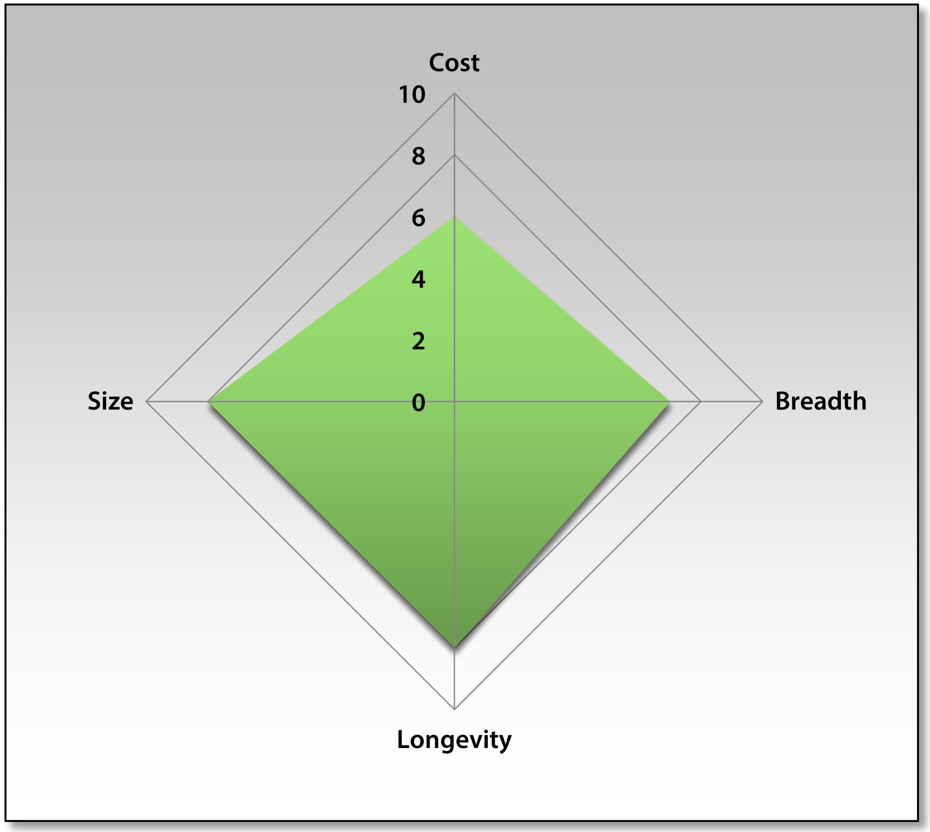
- **Familiar management tools**
  - Windows Explorer!
- **Extra features**
  - Encryption

DiskCache

- **Suitability applied to the decision matrix**
  - Cost – slow, network access + disk IO
  - Breadth – available to any process, any machine with network access
  - Longevity – disk lifespan & network availability
  - Size – available space on disk/SAN

DiskCache

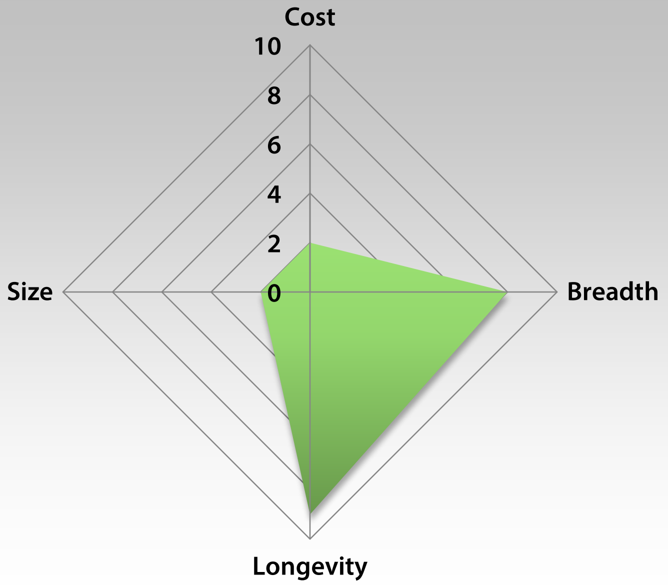
- Good for long-lived items, balanced against performance



Applying the Decision Matrix

- **Vehicle Makes**

Cost	2
Breadth	8
Longevity	9
Size	2

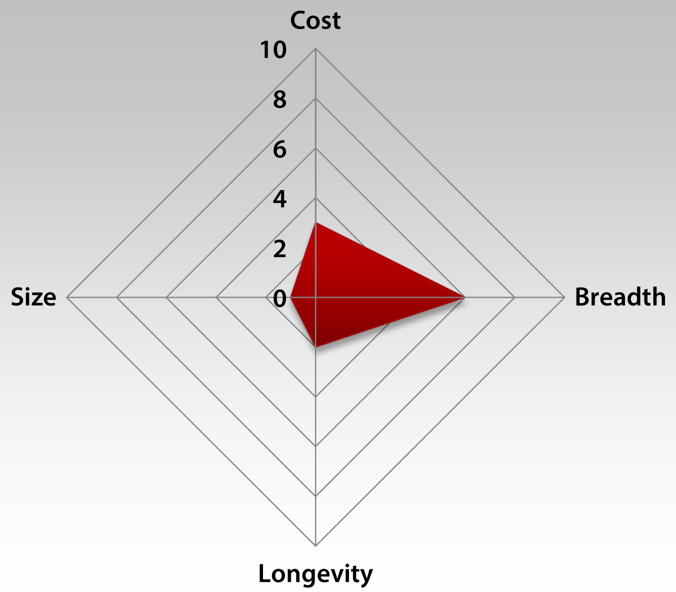


- **Fast, accessible cache, reasonable longevity; size less important**
  - Memcached
  - AppFabric Caching – replicated configuration

Applying the Decision Matrix

- CMS Content

Cost	3
Breadth	6
Longevity	2
Size	1

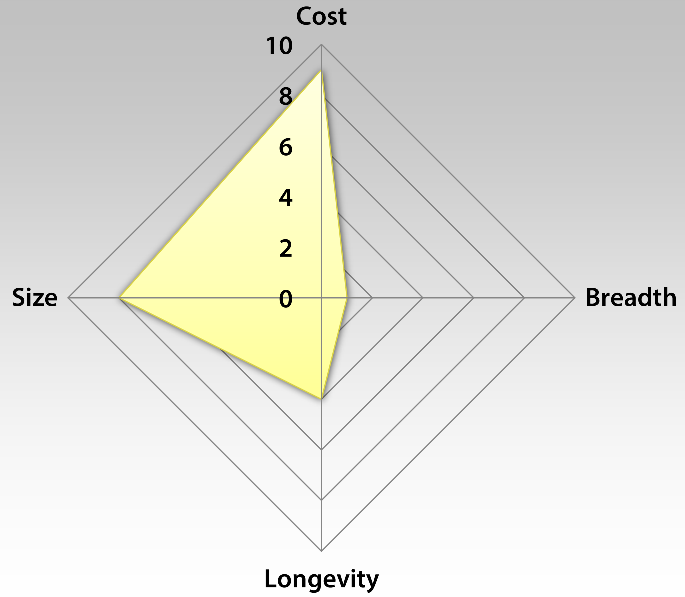


- **Fast, accessible cache; size and longevity less important**
  - MemoryCache
  - NCache Express

Applying the Decision Matrix

- **Quote Prices**

Cost	9
Breadth	1
Longevity	4
Size	8



- **Big, accessible, long-lived cache; speed less important**
  - DiskCache
  - Azure Table Storage

Applying the Decision Matrix

- **Should I have multiple cache stores?**
  - Yes
- **Three levels**
  1. Fast local memory store
  2. Fast remote memory store
  3. (Optional) Large persistent store
- **Which ones?**

	Option 1	*Option 2*
Level 1: fast local	**.NET MemoryCache**	**NCache Express**
Level 2: fast remote	**AppFabric Caching**	**Memcached**
Level 3: large persistent	**DiskCache**	**Azure Table Storage**

Summary

- **Types of remote cache store**
  - Centralized
  - Distributed
  - Replicated
- **Remote caches**
  - Memcached
  - Azure Table Storage
  - DiskCache
- **Applying the Decision Matrix**
  - Selecting the right cache
  - Balance performance with cost
  - Choosing your cache stores

References

- **Memcached**
  - Windows server – 32bit & 64bit
    - http://s3.amazonaws.com/downloads.northscale.com/memcached-win32-1.4.4-14.zip
    - http://s3.amazonaws.com/downloads.northscale.com/memcached-win64-1.4.4-14.zip
  - .NET Client
    - https://github.com/enyim/EnyimMemcached
  - Documentation
    - https://code.google.com/p/memcached/wiki/NewStart
- **Azure Table Storage**
  - Documentation
    - http://msdn.microsoft.com/en-us/windowsserver/ee695849.aspx