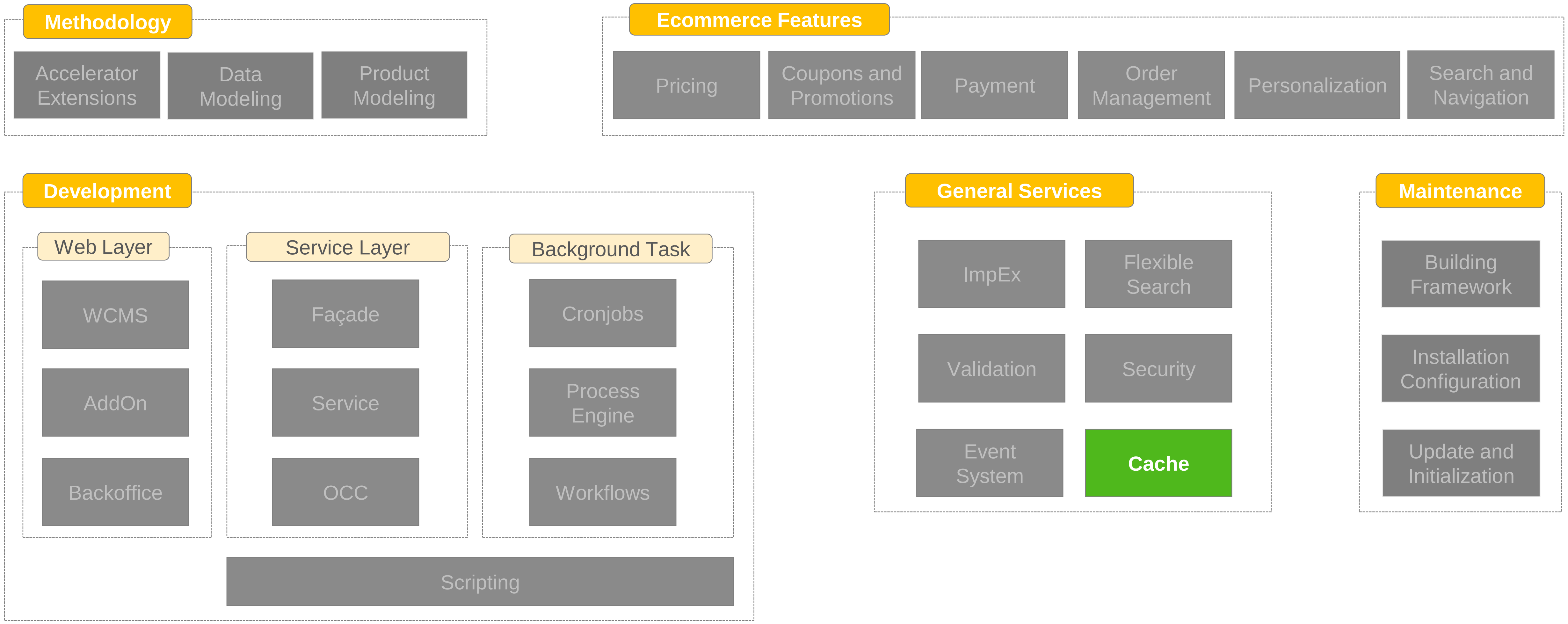
SAP Customer Experience

Cache

SAP Commerce Cloud Developer Training
Part I

THE BEST RUN

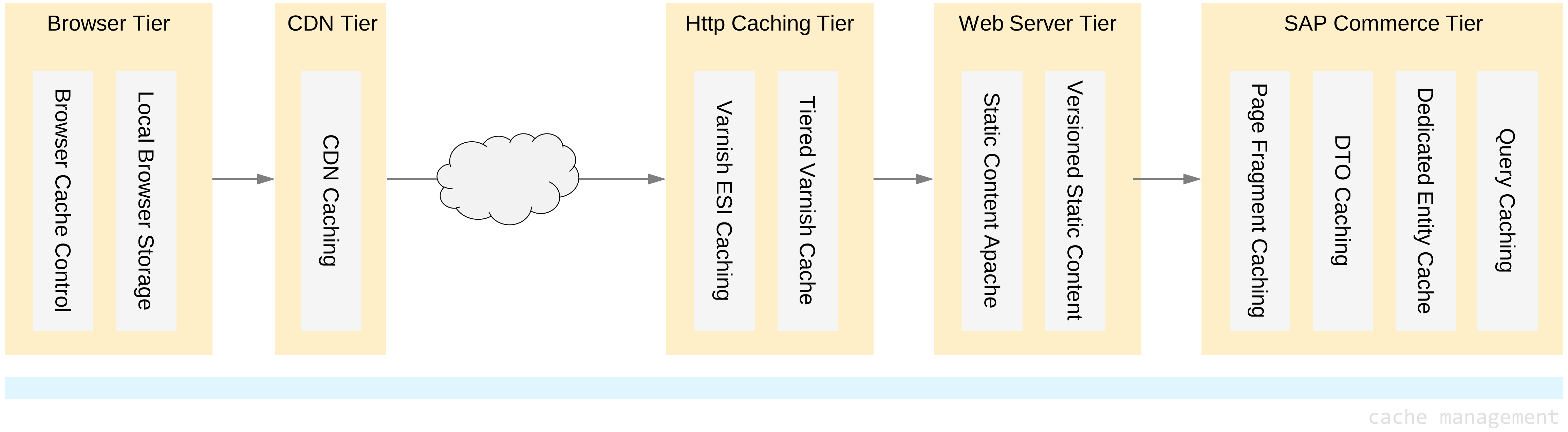
What we will cover in this topic



The Context…

The Cache improves the **performance** of a each server node by reducing the number of database queries. It **transparently reads from and stores** search results, item attributes, and item instances **in memory**.

Caching Tiered Architecture Overview

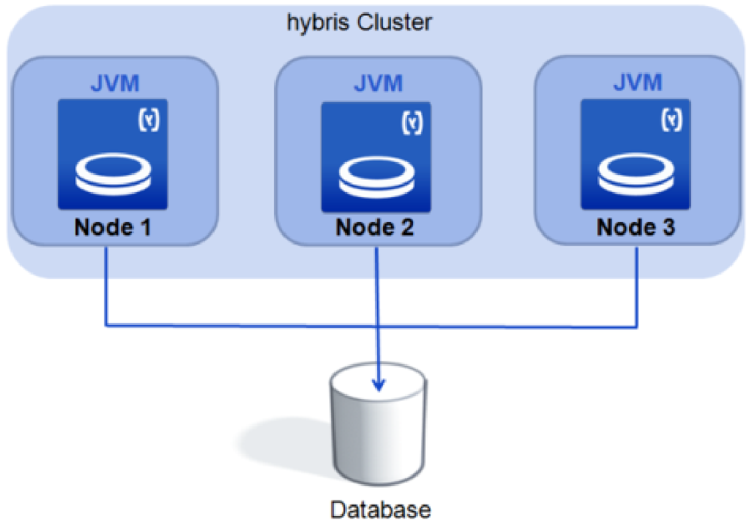


- Should occur as early as possible in the request processing
- Trade off between performance and data freshness
- Centralize management in one place
- Iterative performance tests

Theory

- Regions based cache
  - By default: EHCache implementation of region.
  - Provided: SAP Commerce implementation as an option (for backward compatibility)
- Each region is configurable:
  - What types it’s caching
  - The maximum size
  - The eviction policy
- Each SAP Commerce model is cached
- Each Flexible Search query is cached
  - Hint: avoid FS queries with small differences (e.g. with `new Date()`)
- No master cache server
- Each cluster has its own cache
  - Caches are invalidated by either TCP or UDP network messaging (JGroups)

Cluster communication



- Clustering methods (JGroups leverage udp mcast –recommended–)
- JGroups provides fastest communication (can be used on cloud)
- Settings for JGroups can be observed inside:

```
<path dir="${HYBRIS_BIN_DIR}/platform/…/jgroups-tcp.xml"/>
<path dir="${HYBRIS_BIN_DIR}/platform/…/jgroups-udp.xml"/>
```

When Data Is Cached (and invalidated)

Caching items:

- When calling flexible search or getters that refer to `ComposedTypes`, the underlying data is returned from the cache or, if not yet cached, first retrieved and then written to the cache.
- When calling `modelService.save()`, the cached value is invalidated (and thus removed from the cache)

Caching FlexibleSeach results:

- When executing FlexibleSearch query, like *`SELECT code FROM Product`*, the list of results is cached in the main cache.
- When a product is removed, then its item data and the cached flexible search result for the above query are removed from the cache.

Eviction policies

- **First In, First Out (FIFO)**

  Elements are evicted in the same order as they come in. When a PUT call is made for a new element, and assuming that the maximum limit is reached for the memory store, the element that was placed first (First-In) in the store is the candidate for eviction (First-Out).

- **Least Frequently Used (LFU)**

  For each GET call on the element the number of hits is updated. When a PUT call is made for a new element, and assuming that the maximum limit is reached for the memory store, the element with least number of hits, the Less Frequently Used element, is evicted.

- **Least Recently Used (LRU)**

  The last used timestamp is updated when an element is put into the cache or an element is retrieved from the cache with a GET call.

How Data Is Cached

- Region Cache – configurable
- Standard configuration:

Entities	
Size	100 000
Eviction Strategy	FIFO

Types	
Size	Unlimited
Eviction Strategy	none

Query Results	
Size	20 000
Eviction Strategy	FIFO

Media Items	
Size	500 Mb
Eviction Strategy	LRU

Example of custom cache configuration

Entities	
Size	50 000
Eviction Strategy	LFU

Typesystem	
Size	Unlimited
Eviction Strategy	none

Query Results	
Size	20 000
Eviction Strategy	FIFO

Products	
Size	50 000
Eviction Strategy	LFU

Manual Region	
Size	50 000
Eviction Strategy	LFU

Media Items	
Size	500 Mb
Eviction Strategy	LRU

Default cache region configuration

- To change preconfigured cache region settings, provide new values in local.properties for the preconfigured parameters:

```
# Size of a region that stores all other, non-typesystem and
# non-query objects. Default value is 100000.
regioncache.entityregion.size=50000

# Change eviction policy used by entity region. Possible vales
# are FIFO (default), LFU and LRU.
regioncache.entityregion.evictionpolicy=LRU

# specifies root cache folder for all cached files
media.default.local.cache.rootCacheFolder=cache
# specifies max size of media cache in MB
media.default.local.cache.maxSize=500
```

New cache region configuration

- Create new custom cache region for a particular Type by defining a new CacheRegion component in Spring Global context :

```
<bean name="productCacheRegion"
      class="de.hybris.platform.regioncache.region.impl.EHCacheRegion">
   <constructor-arg name="name" value="productCacheRegion" />
   <constructor-arg name="maxEntries" value="50000" />
   <constructor-arg name="evictionPolicy" value="LFU" />
   <property name="handledTypes">
      <array>
         <value>1</value>
      </array>
   </property>
</bean>
```

References

- Cache in Platform
- https://help.sap.com/viewer/d0224eca81e249cb821f2cdf45a82ace/1905/en-US/8be98ee4866910149df8be0aab4d0b62.html

- Hybris Tuning in ALF

  https://wiki.hybris.com/display/hybrisALF/Hybris+Tuning

- Caching Design in ALF

  https://wiki.hybris.com/display/hybrisALF/Caching

Key POINTS

The Cache is an **end-to-end** solution that will affect every layer of your application

The SAP Commerce service layer cache uses a **region**-based cache solution

You can configure eviction policy, maximum size, and other parameters for each region

When a new query is executed on DB, both the query and the items returned are saved in the cache. Subsequent, identical queries can be immediately satisfied by the cache

When updating/deleting a data item or if the eviction policy is triggered, related queries may be invalidated and affected items will be removed from the cache.

SAP Customer Experience

Thank you.

THE BEST RUN