Hybris Cluster Sizing

Sizing a SAP Hybris Commerce cluster requires an understanding of the cluster architecture, the complexity of the implementation as well as the expected load.

The cluster setup defines the frontend versus backend traffic which is required to differentiate the resource utilization (CPU, RAM) for those different server operations. The complexity of the solution is much harder to define as it is driven by many factors like the modules being used, the quality of code and the configuration optimizations that have been applied to the application server. The number of page request that can be served will be influenced by some of the decisions made in the software architecture, for example, the caching strategies being applied. Given a large number of factors that play a role in the final size, this guide should be seen as a benchmark rather than precise recommendations that can be applied to each implementation.

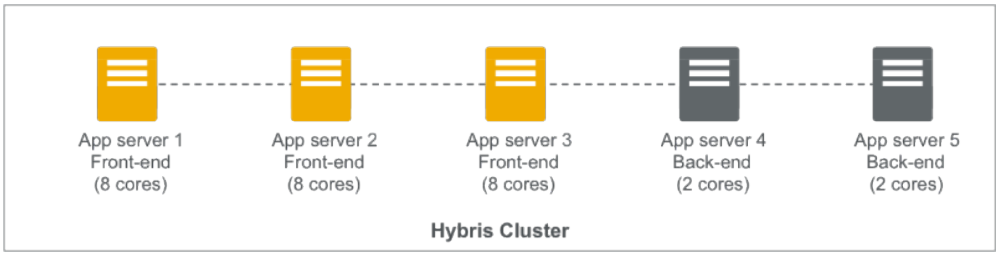
> The recommendations in this practice can be used for sizing estimations, however, performance testing is strongly recommended throughout the project and solution lifecycle to validate and refine the sizing.

Approach

Sizing the cluster requires a definition of the number of cluster nodes as well as the detailed resource allocation per node. The workload for frontend traffic is very different from backend traffic and processes, the sizing estimation should be broken down into its component parts by adopting the recommended cluster setup which appropriately separates nodes by function. The primary interest goes out to the total number of frontend node cores required to serve the peak page load. Based on the total number of front-end cores the cluster nodes can be defined. The nodes are typically equipped with an equal amount of nodes as it will simplify the configuration and optimizations that can be shared by those nodes.

The figure below illustrates a sample cluster setup with a number of front-end nodes as well as backend nodes. The customer facing nodes share 24 cores in total which could also have been served by 5 quad-core nodes. A few larger nodes seem to perform better in general rather than a larger number of small nodes, testing various configurations is however recommended. Having a virtual environment will help to refine the configuration based on performance testing, keep in mind though that CPU virtualization adds varying amounts of overhead depending on the workload and the type of virtualization used.

The example shows 2 smaller nodes for the backend traffic. Separating the backend over multiple nodes is typically done to separate out business users and (CPU intensive) integration tasks.



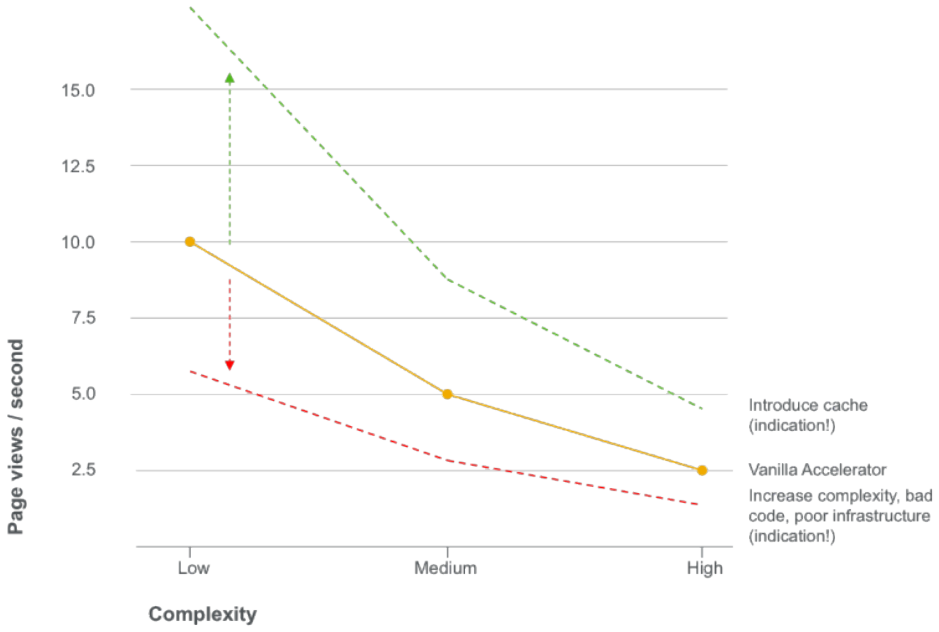
In the following section, we provide recommendations to calculate the CPU, RAM and disk size for each node.

Sizing CPU Cores

To ascertain the initial sizing of the infrastructure some fundamental metrics must be gathered around the expected volume of frontend traffic, the volume of backend processes as well as an indication of system complexity. The complexity of the implementation can have a significant impact on the sizing of the infrastructure due to the extra compute power required to calculate complex requirements. Typical e-commerce complexity can be found in personalized and complicated pricing models, personal product catalogs or complex visibility restrictions on storefronts. B2B commerce implementations are often found to contain such system complexity and are considered to be medium to high range. Once there is a high-level understanding of the complexity the appropriate assumptions can be selected for detailed sizing calculations.

T-shirt sizing can be used to indicate the rough complexity of the implementation, using low, medium and high complexity. The accelerators can be used as a benchmark where the B2C Accelerator is in the low complexity range while the B2B Accelerator is typically attributed to a medium or high complexity range. When using these assumptions for project specific sizing calculations, the effect of customizations on top of the chosen accelerator should also be considered as complexity is often increased which can

potentially reduce performance. To counteract the effect of increased complexity reducing performance, there are a number possible optimizations that can be used on top of the accelerators which will improve performance. Implementing the appropriate cache for each architectural components and tiers will help to improve the number of page impressions served by each node.



Front-End Traffic

Frontend traffic is typically measured using a maximum *page views/second* metric. Using the maximum page views/second metric, we can approximate the number of cores the system will require for storefront traffic. An example for frontend servers is given in the table below. This example illustrates the effect of the system complexity on the number of cores required to meet maximum storefront demand.

Complexity	Workload(PV/s)	Cores
Low	100	roundup(100/10) = 10
Medium	75	roundup(75/5) = 15

There are 2 basic mechanics recommended in order to calculate the number of page views per second, using either the concurrent users or the orders per second as an input. The formulas for those approaches are given below.

$$\text{pv/sec} = \frac{\text{Concurrent users}^{[1]} * \text{number of request}^{[2]}}{3600}$$

$$\text{pv/sec} = \frac{\text{Orders per second}}{\text{number of request}^{[2]} / \text{Conversion rate}^{[3]}}$$

[1] how many open user sessions per hour
[2] average number of requests (=clicks) per customer
[3] number of completed checkouts

The number of page views per second can be calculated if it is not readily available. Metrics that are often monitored by

marketing teams are the number of sessions per hour, number of requests per session per hour and orders per day/hour/second. These metrics are frequently in the case of existing eCommerce sites tracked and monitored using analytics tools such as Google Analytics.

A very high number of SAP Hybris cores from an estimation should be investigated and checked against the instance strategy to ensure that the system can scale within the design, this is especially true for global high volume customers who may need a strategy of multiple SAP Hybris clusters.

Backend Workload

Backend SAP Hybris server workload is driven by a number of factors and is listed in the following table. For large high volume and data intensive SAP Hybris implementations give consideration to separate nodes by function.

Cockpit users	Product cockpit/backoffice, The volume of product enrichment and size of the catalog can have an impact on the sizing of backend nodes. For catalogs that have a large number of products, one should consider using solr product cockpit search solution add-on. Customer Service cockpit - In high usage scenarios use separate dedicated nodes for customer service agents.
Catalog Syncronisation	The number of products in a catalog is a factor in estimating the size of the backend nodes furthermore, the number of product catalogs also influences the amount of synchronization to be performed when a product is updated. The rate at which products are updated and synchronized should also be used as part of your calculation for backend node sizing.
Solr Indexing	Similar to catalog synchronization the number of products and number of indexes influences the sizing of backend nodes that will be performing Solr indexing. The frequency of Solr updates and the number of products that are updated on a daily/hourly basis influence the backend workload.
Business processes (Task Engine)	The frequency and complexity of backend task engine task executions has an impact on resource usage.For example, complex order processing tasks at high frequency will have demands on resource usage and thus will be a contributing factor to the backend workload.
Integrations	Typical integrations are product/price and stock. The volume and frequency of updates will have resource demands on the backend workload and must be factored in as part of the sizing.

Sizing RAM

The amount of RAM available to a SAP Hybris server has a direct impact on how big you can size your Java heap without forcing the operating system to swap memory, which is considered to be a performance overhead. The initial sizing of RAM must consider the memory usage of all running processes on the server leaving enough RAM for the Java heap.

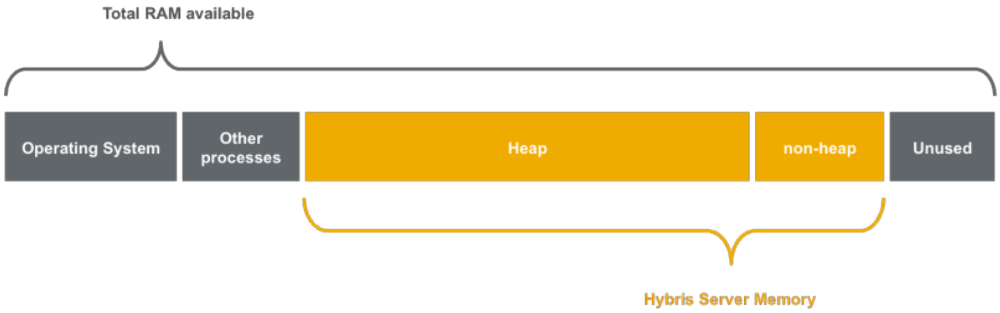
The server memory usage can roughly be broken down into different memory segments:

- Java heap
- Java non-heap memory
- Operating System
- Other running processes. e.g. monitoring tools

The total amount of memory required for each node server is a matter of cumulating the different memory segments. The following assumptions can be used and adjusted according to your project:

- As a guide, the Java heap tends to be sized between 6GB - 16GB of RAM. It is recommended to performance test various heap sizes to find the optimal size and configuration for your project and to leave some additional unused memory for head room. The size of the heap is primarily influenced by the size of the entity and query cache, the number of sessions and the

number of requests served by the hybris server.
- Operating System RAM tends to be between 1GB - 2G
- Other running processes such as monitoring tools tend to be minimal, and a good starting point would be to use 250MB.
- Java non-heap memory is composed of the perm gen in the case of Java 7 and metaspace in Java 8 as well as code cache; a good starting point is using 500MB.



Sizing Disk Space

SAP Hybris server disk space usage can be broken down into the following categories:

- SAP Hybris Platform and SAP Hybris extensions together with custom extensions.
- Media storage of product and content related data.
- Storage of server logs such as console logs.
- Files generated as a result of cronjobs, synchronization and impex and other technical activities that require media items to be created.

Media folders are used for both technical tasks such as cronjobs as well as storage of content and product related media.The media folders are configurable, and a media strategy can be applied either on a global or folder basis for each type of media generated by the SAP Hybris platform. The default media storage is the local media strategy. Depending on the solution choices consideration for each media folder should be evaluated to estimate local disk storage requirements.

The following assumptions can be used and adjusted according to your project:

- SAP Hybris platform, extensions, and custom extensions disk usage between 1.5GB - 2GB.
- Media storage of product and content related images varies from project to project depending on the media strategy employed.
- Logs file have a high variance and range between 1 GB to 100 GB depending on how much is logged and log rotation strategies.

> For transient files created such as log files and media files generated as part of cronjob and synchronization execution, ensure an appropriate data maintenance strategy is planned.