Sizing view

2024-10-08

**Last modified** : 2024-10-08

**Last global review** : *<perform regular global reviews of the view at least once a year while the project is active and mention the date here>*

**Document status** : *<'WIP', 'DRAFT', …​>*

# Introduction

This is the sizing point of view of the project. It determines the hardware required by the project.

The other views of the document are accessible [from here](./README.adoc).

The project glossary is available [here](glossary.adoc). We will not redefine the functional or technical terms used here.

## Reference Documentation

Sizing documentary references

| N ° | Version | Document title / URL | Detail |
| --- | --- | --- | --- |
| *1* | *1.2* | *Report\_benchmark\_xyz.pdf* |  |

# Not ruled

## Points subject to further study

Points subject to further study

| ID | Detail | Status | Subject holder | Deadline |
| --- | --- | --- | --- | --- |
| *1* | *SSD disk capacity* | *WIP* | *XYZ* | *01/01/2022* |

## Assumptions

Give here the structuring assumptions taken for the design

Example:

Assumptions

| ID | Detail |
| --- | --- |
| *1* | *We estimate that 10M people will download our app and use it at least once a day* |

# Constraints

Constraints are the limits applicable to the requirements on the project.

It is interesting to explain them in order to obtain realistic requirements. For example, it would not be realistic to require response times from a web service incompatible with the bandwidth of the underlying network.

## Storage constraints

List here the possible constraints related to the disks

Example: The maximum disk space that can be allocated to a VM is 2 TiB.

## CPU constraints

List here the possible constraints related to the computing power

Example 1: A VM will have a maximum of 10 VCPUs

Example 2: All the pods of the application should not require more than 1 CPU per node.

## Memory constraints

List here the possible constraints related to memory

Example: a pod or a Kubernetes job should not use more than 6 GiB of RAM

## Network constraints

List here the possible network volume constraints used

Example 1: The minimum latency for requests on the WAN between London and Tokyo is 250 ms

Example 2: The Ethernet network of the Datacenter has a bandwidth of 40 Gbps.

# Requirements

It is crucial to collect as much information as possible from production rather than making estimations which often turn out to be far from reality. However, this may not always be possible for new projects. The architects should size the application with a significant margin. The information given here can be used as input to the SLO (Service Level Objective set by the operators).

The following sections dealing with static and dynamic sizing calculations give examples of calculations and elements to consider. Formally, it may be better to use a spreadsheet. For a complex project or one where assumptions may often change, this is essential.

## Static sizing

These are the metrics used to determine the **cumulative** storage volume of the project. Remember to clearly specify the assumptions made for the estimated metrics. It will thus be possible to review them if new business elements appear.

### Metrics

These are measured or estimated business data that will be used as inputs to the calculation of required storage.

| Metric | Description | Measured or Estimated? | Value | Forecast annual increase (%) | Source | Detail/assumptions |
| --- | --- | --- | --- | --- | --- | --- |
| *S1* | *Number of eligible companies* | *Estimated* | *4M* | *+ 1%* | *Government [reference]* | *We consider that AllMyData is applicable only for companies with more than 10 employees* |
| *S2* | *Average size of a PDF* | *Measured* | *40KiB* | *0%* | *Operators* |  |

### Estimated storage needs

List here the storage needs of each component once the application has reached full load (volume at two years for example).

Take into account:

* The size of the databases.
* The size of the files produced.
* The size of the queues.
* The size of the logs.
* …​

Does not take into account:

* The volume linked to the backup: it is managed by the operators.
* The volume of binaries (OS, middleware …​) managed by the operators.
* Archived data which is therefore no longer online.

Also provide an estimate of the annual % increase in volume to allow operators to order or reserve enough disk.

For the sizing calculations, remember to take into account the specificities of the encoding (number of octets by character, by date, by numerical value …​).

For a database, plan the space occupied by the indexes, which is very specific to each application. A (very poor) preliminary estimate is to double the disk space (to be refined later).

Only estimate data whose size is not negligible (several GiB minimum).

1. Example of static sizing of component C:

| Data | Description | Unit size | Number of items at 2 years | Total size | Annual increase |
| --- | --- | --- | --- | --- | --- |
| Table Article | Catalog items | 2 KiB | 100K | 200 MiB | 5% |
| Command Table | Customer orders | 10 KiB | 3M | 26.6 GiB | 10% |
| Logs | Application logs (INFO level) | 200 B | 300M | 56 GiB | 0% (archiving) |

## Dynamic sizing

These are metrics by duration (year, month, hour, …​) and allowing to determine the load applied to the architecture, which will help to size the systems in terms of CPU, bandwidth and performance of disks.

### Metrics

These are the measured or estimated business data that will be used as inputs for the calculation.

| Metric | Description | Measured or Estimated? | Value | Forecast annual increase (%) | Seasonality | Source | Detail/assumptions |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *D1* | *Proportion of users connecting to the service / J* | *Estimated* | *1%* | *+ 5%* | * Constant over the year * Constant over the week * 3 peaks at 20% of the day at 8:00am-9:00am, 11:00am-12:00am and 02:00pm-03:00pm |  | *Users use the application during standard office hours* |

### Estimated load

This involves estimating the number of calls to components and therefore the target throughput (in TPS = Transactions per second) that each of them will have to absorb. A well-sized system should have **average response times of the same order at nominal load and peak**.

Always estimate the **"peak of the peak"**, i.e., the moment when the load will be maximum following the accumulation of all the factors (for example for an accounting system: between 02 pm and 03 pm on a weekday at the end of December).

Do not consider that the load is constant but take into account:

* Daily variations. For a management application with users working during office hours, we typically observe peaks of double the average load at 8-9 a.m., 11-12 a.m. and 2 p.m.-3 p.m. For a consumer Internet application, it will be more at the end of the evening. Again, rely on measurements of similar applications when possible, rather than estimates.
* The elements of seasonality like Christmas for the chocolate industry, Saturday evening for emergency admissions, June for central booking stays etc. The load can then double or even more. This estimate should therefore not be neglected.

If the calculation of the peak for a component at the end of the linking chain is complex (for example, a central IS service exposing referential data and called by many components which each have their own peak), cut the day into time intervals sufficiently small (one hour for example) and the measured or estimated sum of the calls of each caller (batch or GUI) will be calculated over each interval to thus determine the highest cumulative demand.

If the application runs on a PaaS type cloud, the load will be absorbed dynamically, but take care to **estimate the additional cost** and to set consistent consumption limits to respect the budget while ensuring a good level of service.

Example: dynamic volumetric estimation of the REST endpoint GET Detail of the AllMyData application

|  |  |
| --- | --- |
| Maximum rate of users connected at the same time in annual peak | S1 x F1 x 0.2 = 8K / H |
| Average duration of a user session | 15 mins |
| Average number of service calls per session | 10 |
| Charge (Transaction / second) | 8K / 4 x 10/3600 = 5.5 Tps |

For a technical component (such as a database instance) at the end of the linking chain and requested by many services, it is necessary to estimate the number of requests at peak by cumulating the calls from all the clients and to **specify the read/write ratio** when this information is relevant (it is especially important for a database).

The level of detail of the estimate depends on the progress of the application design and the reliability of the assumptions.

In the example below, we already have an idea of ​​the number of requests for each operation. In other cases, we will have to be satisfied with a very broad estimate of the total number of requests to the database and a read/write ratio based on measures from similar applications. No need to go into more detail at this point.

Finally, keep in mind that this is **simply an estimation** yet to be validated during campaigns performance and then in production. Plan a sizing adjustment shortly after the production.

Example: the Oracle BD01 database is used for reading by the REST endpoint GET DetailArticle made from the end-user application and for updating by the POST and PUT calls on DetailArticle endpoint from the supply batch B03 at night between 01:00 and 02:00.

Example estimates number of peak SQL queries to instance BD01 from 01:00 to 02:00 in December

|  |  |
| --- | --- |
| Maximum rate of users logged in at the same time | 0.5% |
| Maximum number of concurrent connected users | 5K |
| Average duration of a user session | 15 mins |
| Average number of calls to the GET DetailArticle endpoint per session | 10 |
| User charge GET DetailArticle (Transaction / second) | (10/15) x 5K / 60 = 55 Tps |
| Number of read and write requests per endpoint call | 2 and 0 |
| Number of daily calls to the POST DetailArticle endpoint from batch B03 | 4K |
| Number of INSERT and SELECT requests per endpoint call | 3 and 2 |
| Daily number of items modified by batch B03 | 10K |
| Number of SELECT and UPDATE queries | 1 and 3 |
| Number of SELECT / sec | 55x2 + 2 x 4K / 3600 + 1 x 10K / 3600 = 115 Tps |
| Number of INSERT / sec | 0 + 3 x 4K / 3600 = 3.4 Tps |
| Number of UPDATE / sec | 0 + 3 x 10K / 3600 = 8.3 Tps |

## Response time requirements

### Response time of GUIs

If the clients access the system via WAN (Internet, VPN, LS, etc.), specify that the **response time requirements are given outside network transit** because it is impossible to commit to the latency and throughput of this type of client.

In the case of LAN access, it is preferable to integrate the network time, as the load testing tools will already take this into account.

The response time objectives are always given with a **statistical tolerance** (90th percentile for example) because reality shows that it is very fluctuating being affected by a large number of factors.

No need to multiply the types of requests (depending on the complexity of the screen, for example) because this type of criterion no longer makes much sense today, particularly for a SPA application).

Example of types of solicitation:

| Type of request | Good level | Medium level | Insufficient level |
| --- | --- | --- | --- |
| *Loading a page* | *<0.5 s* | *<1 s* | *> 2 s* |
| *Business operation* | *<2 s* | *<4 s* | *> 6 s* |
| *Editing, Export, Generation* | *<3 s* | *<6 s* | *> 15 s* |

Example of acceptability of response times:

The level of compliance with response time requirements is good if:

* At least 90% of response times are good.
* At most 2% of response times are insufficient.

Acceptable if:

* At least 80% of response times are good.
* At most 5% of response times are insufficient.

Apart from these values, the application must be optimized and go back to acceptance and then be subjected to load tests again.

### Duration of batch execution

Specify here in what time interval the batch processes should run.

Example 1: The end of the execution of the batches being a prerequisite for the opening of the service to end-users, these first must imperatively end before the end of the batch range defined above.

Example 2: the monthly account consolidation batch B1 must be executed in less than 4 days.

Example 3: the batches and the GUI can operate in competition, there is no strict constraint on the execution time of the batches but to ensure an optimization of the hardware infrastructure, we will favor the night during which the GUI requests are less numerous.

# Target sizing

We give a final sizing to support the static and dynamic sizing and meet the requirements.

## Estimate of resource requirements by technical component

Give here RAM, disk and CPU per instance of technical component deployed (to be refined after performance campaign or MEP).

Example:

Estimation of resource requirements by technical component

| Deployable unit | (V)CPU requirement per instance | Memory requirement per instance (MiB) | Periods of activity | Comments |
| --- | --- | --- | --- | --- |
| *tomcat-batchs1* | *<negligible>* | *1024* | *Every hour, 24/7/365* | *The application server remains started even outside the execution of jobs* |
| *spa* | *<negligible>* | *50* | *24/6, main activity 8am-5pm GMT Mon-Fri* | *SPA Web App, runs in the browser* |
| *bdd-postgresql* | *2* | *2024* | *24/7, main activity 8am-5pm GMT Mon-Fri* | *Postgresql instance* |

## Sizing of machines

This section provides the final sizing of the machines required.

* For Virtual Machines, be careful to check that a VCPU = 1 physical core (and not a thread if hyperthreading is enabled)
* The internal disk concerns the disk necessary for the OS and the binaries. For a physical machine, this is local storage (local SDD, NMVe or HDD disks). For a VM, it can be a local disk on the physical machine running the VM or a SAN.
* The external disk concerns storage on a disk bay (SAN).
* Others external storage deals with distributed file system (NFS, CIFS, WebDav …​) or object storage (Ceph, Swift, Scality, S3, …​)

Sizing of machines

| Zone | Machine type | Number of machines | Number of (V)CPU | Memory (GiB) | Internal disk (GiB) | External SAN disk (GiB) |
| --- | --- | --- | --- | --- | --- | --- |
| Zone 1 | VM application server | 3 | 2 | 4 | 100 | 0 |
| Zone 2 | Physical machine Database | 1 | 2 | 6 | 50 | 1024 (SAN) |

|Nature|Size (Gio)|Type(s) of machine using this share

|NFS (NAS mount) |248 |Physical machine Database

|OpenStack Object Storage ("Ceph") |2000 |VM application server