**HPC Ontology standardisation**

1. **Ontology Background (with respect to ‘OWL’ language)**

**Classes**: Entities in an object-oriented world

**Inheritance**: Inheritance is called “is-a” relationship

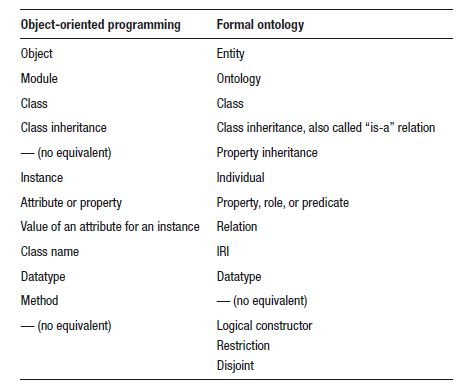
**Disjoints**: Two classes cannot have individuals in common (a given grouping cannot be both x and y)

**Partitions**:class a and b constitute a partition of class AB (when AB is always either a or b)

**Data properties**: properties whose values are data (number, text, date, boolean, etc.)

**Object properties**: properties whose values are entities (ontology individuals)

**Annotation Properties**: properties that can mix data and entities without restriction



* OWL supports **inheritance** between properties, in addition to inheritance between classes
* Each “**data property**” can be configured by specifying: **domain** (the class for which the property is defined), **range** (associated datatype), and its **functional** status (a given individual can have only one value for this property if functional, or if it can have several values)
* Each “**object property**” can be configured by specifying: **domain** (the class for which the property is defined), **range** (associated objects), its **inverse** property (existing relationships when the property is read backward), its **functional** status (a given individual can have only one value for this property if functional, or if it can have several values), its **inverse functional** status (if the inverse property is functional), its **transitive** status (if it is possible to chain the property on several objects), its **symmetric** status (if it can be read indifferently in both directions), its **asymmetric** status (if it is never symmetrical), its **reflexive** status (if it always applies between any object and itself), and its **irreflexive** status (if it is never reflexive)
* We can also add **restrictions** to the classes: *existential restriction* or **some** (the class of individuals who have at least one relation of a certain property with an individual belonging to a certain class), *cardinality restrictions* or **exactly**/**min**/**max** (the class of individuals who have either *exact* or a *minimum* or a *maximum* number of relations of a certain property with an individual belonging to a certain class), *universal restriction* or **only** (the class of individuals who only have a relation of a certain property with one or more individuals belonging to a certain class), and *value restriction* or **value** (the class of individuals who have a certain value for a certain property; sometimes called *role-filler*)
* OWL allows the use of **logical operators** as constructors: **logical AND** or intersection (individuals belonging to several classes at the same time), **logical OR** or union (individuals belonging to a class among several), and **logical NOT** or compliment (individuals who do not belong to a given class)
* OWL also allows using a formal equivalence definition via an **equivalence relation** (the defined classes allow reclassifying individuals during automatic reasoning)

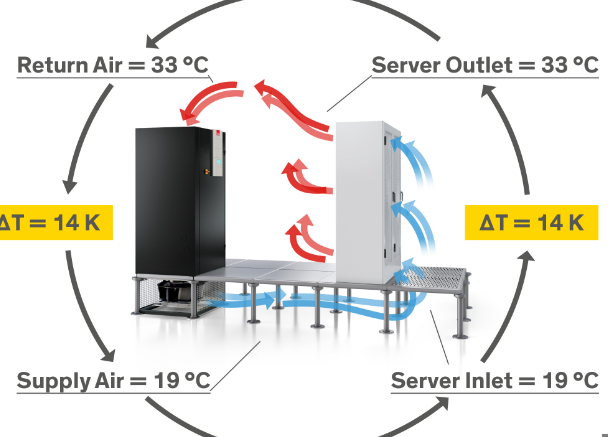
1. **CINECA’s Marconi100 HPC cluster dataset**

* Modelling data extracted from IPMI (**Intelligent Platform Management Interface**) plugin (https://gitlab.com/ecs-lab/exadata/-/blob/main/documentation/plugins/ipmi.md)

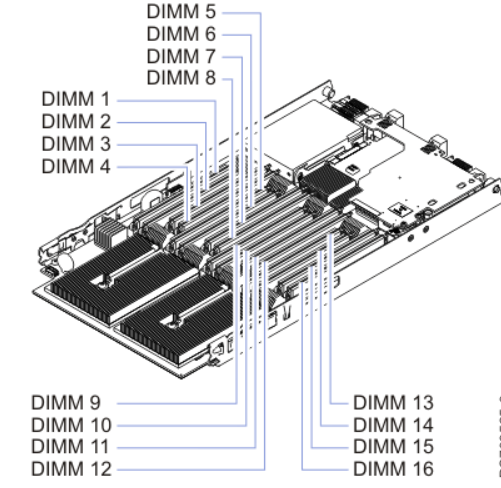
| **metric (description)** | **unit** | **value type** | **sampling period** |
| --- | --- | --- | --- |
| ambient (Temperature at the node inlet) | measured in °C | values are of type float | sampled at 20s from each node |
| dimmX\_temp (Temperature of DIMM module X. X=0..15) | measured in °C | values are of type int | sampled at 20s from each node |
| fanX\_Y (Speed of the Fan Y in module X. X=0..3, Y=0,1) | measured in revolutions (RPM) | values are of type int | sampled at 20s from each node |
| fan\_disk\_power (Temperature at the node inlet) | measured in Watts | values are of type int | sampled at 20s from each node |
| gpuX\_core\_temp (Temperature of the core for the GPU id X. X=0,1,3,4) | measured in °C | values are of type int | sampled at 20s from each node |
| gpuX\_mem\_temp (Temperature of the memory for the GPU id X. X=0,1,3,4) | measured in °C | values are of type int | sampled at 20s from each node |
| gv100cardX (Nvidia Quadro GV100 Card for the GPU id X. X=0,1,3,4) | measurement is unspecified | values are of type int | sampled at 20s from each node |
| pX\_coreY\_temp (Temperature of core Y in the CPU socket X. X=0..1, Y=0..23) | measured in °C | values are of type int | sampled at 20s from each node |
| pX\_io\_power (Power consumption for the I/O subsystem for the CPU socket X. X=0..1) | measured in Watts | values are of type int | sampled at 20s from each node |
| pX\_mem\_power (Power consumption for the memory subsystem for the CPU socket X. X=0..1) | measured in Watts | values are of type int | sampled at 20s from each node |
| pX\_power (Power consumption for the CPU socket X. X=0..1) | measured in Watts | values are of type int | sampled at 20s from each node |
| pX\_vdd\_temp (Temperature of the voltage regulator for the CPU socket X. X=0..1) | measured in °C | values are of type int | sampled at 20s from each node |
| pcie (Temperature at the PCIExpress slots) | measured in °C | values are of type float | sampled at 20s from each node |
| psX\_input\_power (Power consumption at the input of power supply X. X=0..1) | measured in Watts | values are of type int | sampled at 20s from each node |
| psX\_input\_voltag (Voltage at the input of power supply X. X=0..1) | measured in Volts | values are of type int | sampled at 20s from each node |
| psX\_output\_curre (Current at the output of power supply X. X=0..1) | measured in Amps | values are of type int | sampled at 20s from each node |
| psX\_output\_volta (Voltage at the output of power supply X. X=0..1) | measured in Volts | values are of type float | sampled at 20s from each node |
| total\_power (Total node power consumption) | measured in Watts | values are of type int | sampled at 20s from each node |

* Dataset metric representation in ontology framework (conceptually, but could not be modelled): 

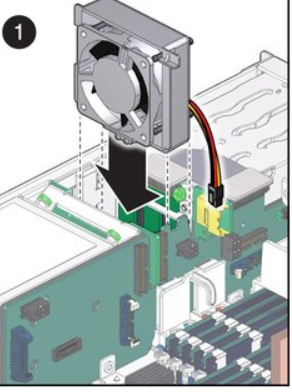
1. Ambient: Server “some-node” has\_member “server-inlet” that records temperature “some-value” generated at time “some-timestamp” (metadata to be reconsidered for inclusion – year\_month, plugin, metric)



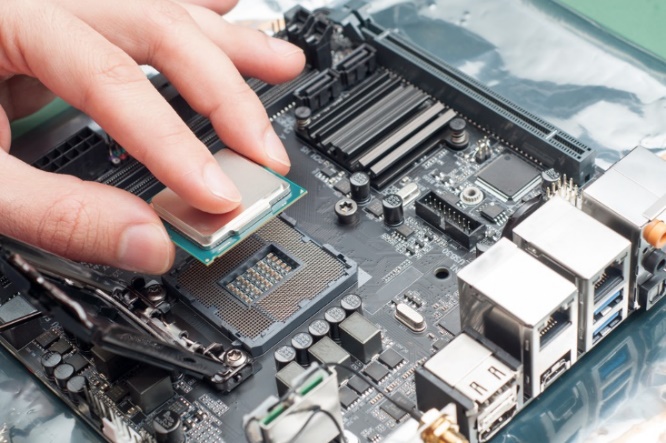
1. dimmX\_temp: Server “some-node” has\_DIMM/Dual\_In-line\_Memory\_Module “some-dimm-value” that records temperature “some-value” generated at time “some-timestamp”



1. fanX\_Y: Server “some-node” has\_drive\_shelf that has\_fan\_module “fan-module-value”, which has\_fan “fan-value”, which has speed “some-value” generated at time “some-timestamp”
2. fan\_disk\_power: Server “some-node” has\_disk\_fan, which consumes\_power “power-value” generated at time “some-timestamp” (metadata to be reconsidered for inclusion – year\_month, plugin, metric)



1. gpuX\_core\_temp: Server “some-node” has\_gpu “some-gpu-value” which has gpu-core, which records temperature “some-value” generated at time “some-timestamp”
2. gpuX\_mem\_temp: Server “some-node” has\_gpu “some-gpu-value” which has gpu\_memory\_size\_per\_node, which records temperature “some-value” generated at time “some-timestamp”
3. gv100cardX: Server “some-node” has\_graphics\_card “NVIDIA Quadro GV100” which has\_value “some-value” generated at time “some-timestamp”



1. pX\_coreY\_temp: Server “some-node” has\_cpu\_socket “cpu-socket-value” which has\_cpu\_core “cpu-core-value”, which records temperature “some-value” generated at time “some-timestamp”
2. pX\_io\_power: Server “some-node” has\_cpu\_socket “cpu-socket-value” which has\_I/O\_subsystem/ has\_Input/Output\_subsystem, which consumes\_power “some-value” generated at time “some-timestamp”
3. pX\_mem\_power: Server “some-node” has\_cpu\_socket “cpu-socket-value” which has\_memory\_subsystem, which consumes\_power “some-value” generated at time “some-timestamp”
4. pX\_power: Server “some-node” has\_cpu\_socket “cpu-socket-value” which consumes\_power “some-value” generated at time “some-timestamp”
5. pX\_vdd\_temp: Server “some-node” has\_cpu\_socket “cpu-socket-value” which has\_voltage\_regulator, which records temperature “some-value” generated at time “some-timestamp”
6. pcie: Server “some-node” has PCI express slot, which records temperature “some-value” generated at time “some-timestamp” (metadata to be reconsidered for inclusion – year\_month, plugin, metric)
7. psX\_input\_power: Server “some-node” has\_power\_supply\_input\_end “power-supply-number-value” which consumes\_power “some-value” generated at time “some-timestamp”
8. psX\_input\_voltag: Server “some-node” has\_power\_supply\_input\_end “power-supply-number-value” which has\_voltage “some-value” generated at time “some-timestamp”
9. psX\_output\_curre: Server “some-node” has\_power\_supply\_output\_end “power-supply-number-value” which has\_current “some-value” generated at time “some-timestamp”
10. psX\_output\_volta: Server “some-node” has\_power\_supply\_output\_end “power-supply-number-value” which has\_voltage “some-value” generated at time “some-timestamp”
11. total\_power: Server “some-node” consumes\_total\_power “some-value” generated at time “some-timestamp” (metadata to be reconsidered for inclusion – year\_month, plugin, metric)

**Alternative Approach** (Representation of time-series observations based on the literature “An Ontological Representation of Time Series Observations on the Semantic Sensor Web”) Suggests usage of three important classes as listed below:

* Observation class (an act of observing a property or phenomenon with the intent to produce an estimate of the value of the property).

**->** Some of the relationships for observations are:

**featureOfInterest** (representation of the object being observed)

**observedProperty** (the phenomenon for which the observation result provides an estimate of its value)

**samplingTime/generatedAtTime** (the time when the phenomenon was measured)

**~~observationLocation~~** (location of an observation event, could be associated with the location of the sensor when an observation occurred) // Not-Used

**result/value** (an estimate of the value of a property generated using a known procedure)

**~~procedure~~** (description of process used to generate the result) // Not-Used

**memberOf** (a relation to a set of observations of observation collection)

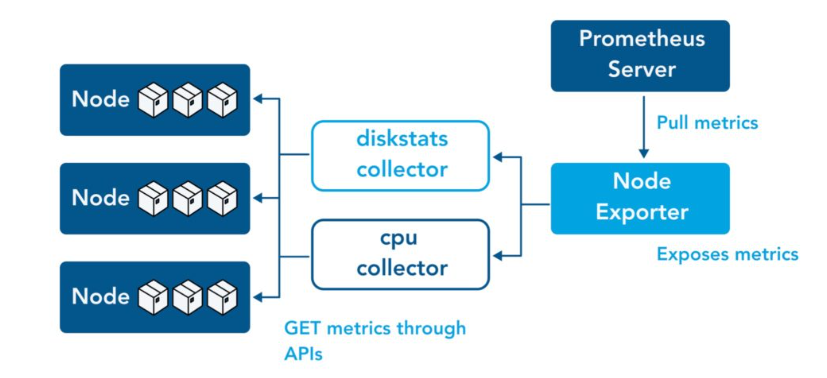
* ~~Observation Collection class (composes of a set of member observations), which has the relationship member (a relation from an observation collection to a constituent observation)~~ *//* ***Not-Used***
* ~~Time Series Observation class (a specialised type of observation collection, which is also considered as a type of observation)~~  *//* ***Not-Used***
* Alternative ontological modelling based on this approach (conceptual representation modelling works): 

1. **Ambient**: Server “some-node” has featureOfInterest “ServerInlet” that has observedProperty “Temperature” that hasTemperatureValue = “some-value” generatedAtTime = “some-timestamp” (metadata to be reconsidered for inclusion – year\_month, plugin, metric?)
2. **dimmX\_temp**: Server “some-node” has featureOfInterest “DIMM/Dual\_In-line\_Memory\_Module” – DIMM1,2,…n that has observedProperty “Temperature” that hasTemperatureValue = “some-value” generatedAtTime = “some-timestamp”
3. **fanX\_Y**: Server “some-node” has featureOfInterest “DriveShelfFanModule” – “fan-module-X”, which has featureOfInterest “Fan” – “fan-value-Y”, that has observedProperty “Speed” that hasSpeedValue = “some-speed-value” generated at time = “some-timestamp”
4. **fan\_disk\_power**: Server “some-node” has featureOfInterest “DiskFan”, that has observedProperty “Power” that hasPowerValue = “some-power-value” generated at time = “some-timestamp” (metadata to be reconsidered for inclusion – year\_month, plugin, metric?)
5. **gpuX\_core\_temp**: Server “some-node” has featureOfInterest “GPU” – GPU1,2,…n, which has featureOfInterest “GPUCore” that has observedProperty “Temperature” that hasTemperatureValue = “some-value” generatedAtTime = “some-timestamp”
6. **gpuX\_mem\_temp**: Server “some-node” has featureOfInterest “GPU” – GPU1,2,…n, which has featureOfInterest “GPUMemory” that has observedProperty “Temperature” that hasTemperatureValue = “some-value” generatedAtTime = “some-timestamp”
7. **gv100cardX**: Server “some-node” has featureOfInterest “GraphicsCard”, that has observedProperty “NVIDIAQuadroGV100” that hasValue = “some-value” generated at time = “some-timestamp”
8. **pX\_coreY\_temp**: : Server “some-node” has featureOfInterest “CPUSocket” – “cpu-socket-X”, which has featureOfInterest “CPUSocketCore” – “cpu-socket-core-Y”, that has observedProperty “Temperature” that hasTemperatureValue = “some-temperature-value” generated at time = “some-timestamp”
9. **pX\_io\_power**: Server “some-node” has featureOfInterest “CPUSocket” – CPUSocket1,2,…n, which has featureOfInterest “InputOutputSubsystem” that has observedProperty “Power” that hasPowerValue = “some-value” generatedAtTime = “some-timestamp”
10. **pX\_mem\_power**: Server “some-node” has featureOfInterest “CPUSocket” – CPUSocket1,2,…n, which has featureOfInterest “MemorySubsystem” that has observedProperty “Power” that hasPowerValue = “some-value” generatedAtTime = “some-timestamp”
11. **pX\_power**: Server “some-node” has featureOfInterest “CPUSocket” – CPUSocket1,2,…n, that has observedProperty “Power” that hasPowerValue = “some-value” generatedAtTime = “some-timestamp”
12. **pX\_vdd\_temp**: Server “some-node” has featureOfInterest “CPUSocket” – CPUSocket1,2,…n, which has featureOfInterest “VoltageRegulator” that has observedProperty “Temperature” that hasTemperatureValue = “some-value” generatedAtTime = “some-timestamp”
13. **pcie**: Server “some-node” has featureOfInterest “PCIExpressSlot”, that has observedProperty “Temperature” that hasTemperatureValue = “some-temperature-value” generated at time = “some-timestamp” (metadata to be reconsidered for inclusion – year\_month, plugin, metric?)
14. **psX\_input\_power**: Server “some-node” has featureOfInterest “PowerSupplyInput” – PSInput1,2,…n that has observedProperty “Power” that hasPowerValue = “some-value” generatedAtTime = “some-timestamp”
15. **psX\_input\_voltag**: Server “some-node” has featureOfInterest “PowerSupplyInput” – PSInput1,2,…n that has observedProperty “Voltage” that hasVoltageValue = “some-value” generatedAtTime = “some-timestamp”
16. **psX\_output\_curre**: Server “some-node” has featureOfInterest “PowerSupplyOutput” – PSInput1,2,…n that has observedProperty “Current” that hasCurrentValue = “some-value” generatedAtTime = “some-timestamp”
17. **psX\_output\_volta**: Server “some-node” has featureOfInterest “PowerSupplyOutput” – PSInput1,2,…n that has observedProperty “Voltage” that hasVoltageValue = “some-value” generatedAtTime = “some-timestamp”
18. **total\_power**: Server “some-node” has observedProperty “Power” that hasPowerValue = “some-value” generatedAtTime = “some-timestamp” (metadata to be reconsidered for inclusion – year\_month, plugin, metric?)
19. **SURF’s LISA cluster dataset**

Prometheus/Node monitor (<https://docs.splunk.com/Observability/gdi/monitors-monitoring/prometheus-node.html>), and own web-search reference from <https://iphostmonitor.com/mib/rfc1213.html>, <https://github.com/prometheus/node_exporter/blob/master/collector/fixtures/e2e-output.txt> and <https://grafana.com/oss/prometheus/exporters/node-exporter/assets/node_exporter_sample_scrape.txt>

* It must be noted that aggregation operators such as sum, min, max and avg have been used in several metrics in the dataset, before exporting these metrics to one of the available Prometheus Exporters (Splunk).

[Image source: <https://www.opsramp.com/guides/prometheus-monitoring/prometheus-node-exporter/>]; some of the other exporters apparently used to generate the dataset are Nvidia GPU exporter, etc.



* Metrics are listed below:  
  ----------------------------------

**Prometheus metrics**   
id: Measurement identifier; int

timestamp: timestamp of the event/measurement (every 30 seconds); datetime

node: name of the node; string

node\_time\_seconds: System time in seconds since epoch (1970)

node\_load15: 15m load average

surfsara\_power\_usage: power usage for the datacentre

up: whether the node is up or not

node\_netstat\_Tcp\_OutSegs: Statistic TcpOutSegs (The total number of segments sent, including those on current connections but excluding those containing only retransmitted octets)

node\_netstat\_Tcp\_InErrs: Statistic TcpInErrs (The total number of segments received in error (e.g., bad TCP checksums))

node\_context\_switches\_total: Total number of context switches

node\_load5: 5m load average

node\_load1: 1m load average

node\_memory\_Active\_bytes: Memory information field Active\_bytes

node\_netstat\_Tcp\_RetransSegs: Statistic TcpRetransSegs (The total number of segments retransmitted - that is, the number of TCP segments transmitted containing one or more previously transmitted octets)

node\_netstat\_Udp\_InErrors: Statistic UdpInErrors (The number of received UDP datagrams that could not be delivered for reasons other than the lack of an application at the destination port)

node\_memory\_Dirty\_bytes: Memory information field Dirty\_bytes (Contains the amount of dirty memory at which a process generating disk writes will itself start writeback)

surfsara\_ambient\_temp: Temperature of the datacenter

node\_netstat\_Icmp\_InMsgs: Statistic IcmpInMsgs (The number of ICMP messages which the node received)

node\_netstat\_Udp\_InDatagrams: Statistic UdpInDatagrams (The total number of UDP datagrams delivered to UDP users of the node)

node\_intr\_total: Total number of interrupts serviced

node\_netstat\_Tcp\_InSegs: Protocol Tcp statistic InSegs (The total number of segments received, including those received in error; this count includes segments received on currently established connections)

node\_memory\_Percpu\_bytes: Memory information field Percpu\_bytes

node\_boot\_time\_seconds: Node boot time, in unixtime

node\_netstat\_Udp\_OutDatagrams: Statistic UdpOutDatagrams (The total number of UDP datagrams sent from this entity)

node\_netstat\_Icmp\_InErrors: Statistic IcmpInErrors (The number of ICMP messages which the entity received but determined as having ICMP-specific errors (bad ICMP checksums, bad length, etc.))

node\_procs\_blocked: Number of processes blocked waiting for I/O to complete

node\_netstat\_Icmp\_OutMsgs: Statistic IcmpOutMsgs (The total number of ICMP messages which this entity attempted to send; note that this counter includes all those counted by icmpOutErrors)

node\_memory\_MemFree\_bytes: Memory information field MemFree\_bytes

node\_procs\_running: Number of processes in runnable state

node\_forks\_total: Total number of forks

node\_hwmon\_temp\_celsius-min: Temperature of the node; Celsius (min)

node\_hwmon\_temp\_celsius-mean: Temperature of the node; Celsius (mean)

node\_hwmon\_temp\_celsius-max: Temperature of the node; Celsius (max)

node\_filesystem\_avail\_bytes-sum: Filesystem space available to non-root users in bytes (sum)

node\_filesystem\_files-sum: Filesystem total file nodes (sum)

node\_network\_transmit\_bytes\_total-sum: Network device statistic transmit\_bytes (sum)

node\_filesystem\_device\_error-sum: Whether an error occurred while getting statistics for the given device (sum)

node\_disk\_written\_bytes\_total-sum: The total number of bytes written successfully (sum)

node\_filesystem\_free\_bytes-sum: Filesystem free space in bytes (sum)

node\_thermal\_zone\_temp-min: The thermal zone temperature of the node (min); Celsius

node\_thermal\_zone\_temp-mean: The thermal zone temperature of the node (mean) ; Celsius

node\_thermal\_zone\_temp-max: The thermal zone temperature of the node (max) ; Celsius

node\_disk\_read\_bytes\_total-sum: The total number of bytes read successfully (sum)

nvidia\_gpu\_memory\_used\_bytes-sum: Memory used by the GPU device in bytes (sum)

nvidia\_gpu\_temperature\_celsius-min: Temperature of the GPU device in Celsius (min)

nvidia\_gpu\_temperature\_celsius-mean: Temperature of the GPU device in Celsius (mean)

nvidia\_gpu\_temperature\_celsius-max: Temperature of the GPU device in Celsius (max)

node\_arp\_entries-sum: ARP entries by device (sum)

nvidia\_gpu\_fanspeed\_percent-min: Fanspeed of the GPU device as a percent of its maximum (min)

nvidia\_gpu\_fanspeed\_percent-mean: Fanspeed of the GPU device as a percent of its maximum (mean)

nvidia\_gpu\_fanspeed\_percent-max: Fanspeed of the GPU device as a percent of its maximum (max)

node\_filesystem\_files\_free-sum: Filesystem total free file nodes (sum)

nvidia\_gpu\_power\_usage\_milliwatts-min: Power usage of the GPU device in milliwatts (min)

nvidia\_gpu\_power\_usage\_milliwatts-sum: Power usage of the GPU device in milliwatts (sum)

nvidia\_gpu\_power\_usage\_milliwatts-mean: Power usage of the GPU device in milliwatts (mean)

nvidia\_gpu\_power\_usage\_milliwatts-max: Power usage of the GPU device in milliwatts (max)

node\_filesystem\_size\_bytes-sum: Filesystem size in bytes (sum)

node\_disk\_writes\_completed\_total-sum: The total number of writes completed successfully (sum)

nvidia\_gpu\_duty\_cycle-min: Percent of time over the past sample period during which one or more kernels were executing on the GPU device (min)

nvidia\_gpu\_duty\_cycle-mean: Percent of time over the past sample period during which one or more kernels were executing on the GPU device (mean)

nvidia\_gpu\_duty\_cycle-max: Percent of time over the past sample period during which one or more kernels were executing on the GPU device (max)

node\_network\_transmit\_packets\_total-sum: Network device statistic transmit\_packets (sum)

node\_udp\_queues-sum: Number of allocated memory in the kernel for UDP datagrams in bytes (sum)

node\_network\_receive\_bytes\_total-sum: Network device statistic receive\_bytes (sum)

node\_network\_receive\_packets\_total-sum: Network device statistic receive\_packets (sum)

node\_network\_receive\_multicast\_total-sum: Network device statistic receive\_multicast (sum)

node\_disk\_io\_now-sum: The number of I/Os currently in progress (sum)

node\_rapl\_package\_joules\_total-sum: Current RAPL (Running Average Power Limit) package value in joules (sum)

node\_network\_receive\_drop\_total-sum: Network device statistic receive\_drop (sum)

**SLURM metrics**

Id: Job ID (int)

start\_date: Start date of job (datetime)

end\_date: End date of job (datetime)

node: Rack and node of the job, maybe multiple (string)

nodetypes: Type of nodes (string)

numnodes: Number of nodes used for the job (int)

numcores: Number of cores used for the job (int)

sharednode: Description unavailable

submit: Job submission time (timestamp)

start: Job start time (timestamp)

end: Job end time (timestamp)

state: Job end state (string)

exitcode: Job exit code

reservation: Information unavailable

partprepaid: Information unavailable

**Custom-derived attributes in SLURM metrics**

node\_num\_in\_rack: node number in a given rack (derived from ‘node’ column)

rack: rack number (derived from ‘node’ column)

**Ontology relationship**:

**Prometheus**

Cluster <HPC-Cluster> hasMonitoringSystem <MonitoringSystem=Prometheus> which hasMetricsExporter <MetricsExporter=PrometheusNodeExporter> that exportsMonitoringMetrics = <MonitoringMetrics>

:that hasMeasurementIdentifier <id> generatedAtTime = “some-timestamp”

: and measuresValueOfThing Site → “DataCentre” that has:  
 -> observedProperty “AmbientTemperature” that hasTemperatureValue = “surfsara\_ambient\_temp”

-> observedProperty “PowerUsage” that hasPowerUsageValue = “surfsara\_power\_usage”

: and measuresValueOfThing Server = <node\_num\_in\_rack> atLocation Rack = <rack>, which has:

-> observedProperty “DeviceStatus” that hasUpValue = “up”

-> observedProperty “TimeSinceEpoch” that hasTimeSinceEpochValue = “node\_time\_seconds” (measured in seconds since epoch (1970))

-> hasMember “Motherboard” which has featureOfInterest “ThermalZone” that has observedProperty “Temperature” that has TemperatureValue = “some-value” (has aggregated values as max. min, and mean)

-> featureOfInterest MetricCollector = “SystemStatisticsCollector” that has observedProperty

(i) “ProcessesBlocked” that hasNumberOfProcessesBlockedValue = “some-value”

(ii) “ProcessesRunning” that hasNumberOfProcessesRunningValue = “some-value”

-> featureOfInterest MetricCollector = “MemoryStatisticsCollector” that has observedProperty

(i) “ActiveBytes” that hasActiveBytesValue = “some-value”

(ii) “DirtyBytes” that hasDirtyBytesValue = “some-value”

(iii) “MemFreeBytes” that hasTotalFreeRAMValue = “some-value”

(iv) “PerCPUBytes” that hasMemoryAllocatedToPerCPUAllocatorValue = “some-value”

-> featureOfInterest MetricCollector = “DiskIOStatisticsCollector” that has observedProperty

(i) “NumberOfIOs” that hasNumberOfIOsInProgressValue = “some-value” (has aggregated values as sum)

(ii) “TotalWrites” that hasTotalNumberOfWritesCompletedValue = “some-value” (has aggregated values as sum)

(iii) “ReadBytes” that hasTotalNumberOfReadBytesValue = “some-value” (has aggregated values as sum)

(iv) “WrittenBytes” that hasTotalNumberOfWrittenBytesValue = “some-value” (has aggregated values as sum)

-> featureOfInterest MetricCollector = “HardwareMonitor” that has observedProperty “NodeTemperature” that hasNodeTemperatureValue = “some-value” (measured in Celsius; has aggregated values as max, min and mean)

-> featureOfInterest MetricCollector = “CPULoadCollector” that has observedProperty “CPULoadAverage” that hasCPULoadAverageValue = “some-value” (measured in different time-intervals of 1, 5, and 15 minutes)

-> featureOfInterest MetricCollector = “ARPMetricCollector” that has observedProperty “ARPEntries” that hasARPEntriesValue = “some-value”

-> featureOfInterest MetricCollector = “BootTimeMetricCollector” that has observedProperty “BootTime” that hasBootTimeValue = “some-value” (measured in seconds)

-> featureOfInterest MetricCollector = “EnergyUsageMetricCollector” that has observedProperty “RunningAveragePowerLimit” that hasRAPLPackageValue = “some-value” (measured in Joules; has aggregated values as sum)

-> featureOfInterest MetricCollector = “UDPQueuesCollector” that has observedProperty “UDPQueues” that hasAllocatedMemoryForUDPDatagramsValue = “some-value”

-> featureOfInterest MetricCollector = “VirtualMemoryStatisticsCollector” that has observedProperty

(i) “ContextSwitches” that hasTotalContextSwitchesValue = “some-value”

(ii) “Forks” that hasTotalForksValue = “some-value”

(iii) “Interrupts” that hasTotalInterruptsServicedValue = “some-value”

-> featureOfInterest MetricCollector = “FileSystemMetricCollector” that has observedProperty

(i) “FilesFree” that hasTotalFreeFileNodesValue = “some-value” (has aggregated values as sum)

(ii) “Files” that hasTotalFileNodesValue = “some-value” (has aggregated values as sum)

(iii) “FreeSpace” that hasTotalFreeSpaceValue = “some-value” (measured in bytes; has aggregated values as sum)

(iv) “DeviceErrorReport” that hasDeviceErrorValue = “some-value” (has aggregated values as sum)

(v) “FileSystemSpace” that hasAvailableFileSystemSpaceToNonRootUsersValue = “some-value” (measured in bytes; has aggregated values as sum)

(vi) “FileSystemSize” that hasFileSystemSizeValue = “some-value” (measured in bytes; has aggregated values as sum)

-> hasMember “GPU” which has featureOfInterest “GPUDevice” that has:

:observedProperty “FanSpeed” that hasFanSpeedValue = “some-value” (measured as a percent of its maximum; has aggregated values as max, min and mean),   
: observedProperty “Temperature” that hasTemperatureValue = “some-value” (measured in Celsius; has aggregated values as max, min and mean),   
: observedProperty “PowerUsage” that hasPowerUsageValue = “some-value” (measured in milliwatts; has aggregated values as max, min, mean and sum),   
:observedProperty “DutyCycle” that hasDutyCycleValue = “some-value” (measured as percent of time over the past sample period during which one or more kernels were executing on the GPU device; has aggregated values as max, min and mean)  
:observedProperty “Memory” that hasMemoryUsedValue = “some-value” (measured in bytes; has aggregated values as sum)

-> featureOfInterest “NetworkInterface” which has observedProperty “NetworkInterfaceMetrics”:

: hasReceiveBytesTotal = “some-value”

: hasReceivePacketsTotal = “some-value”

: hasReceiveMulticastPacketsTotal = “some-value”

: hasReceiveDroppedPacketsTotal = “some-value”

: hasTransmitBytesTotal = “some-value”

: hasTransmitPacketsTotal = “some-value”

-> featureOfInterest “KernelNetworkSubsystem”:

(i) that has observedProperty “ICMPMetrics” that

:hasICMPInErrorsMessagesValue = “some-value”

:hasICMPInMessagesValue = “some-value”

:hasICMPOutMessagesValue = “some-value”

(ii) that has observedProperty “TCPMetrics” that

:hasTCPInErrorsSegmentsValue = “some-value”

:hasTCPInSegmentsValue = “some-value”

:hasTCPOutSegmentsValue = “some-value”

:hasTCPRetransmittedSegmentsValue = “some-value”

(iii) that has observedProperty “UDPMetrics” that

:hasUDPInDatagramsValue = “some-value”

:hasUDPOutDatagramsValue = “some-value”

:hasUDPInErrorsDatagramsValue = “some-value”

**SLURM (Simple Linux Utility for Resource Management)**

Cluster <HPC-Cluster> hasResourceManager ‘SLURM’ which hasJobScheduler <Slurm-Workload-Manager>, that managesJob <Job>:

* that hasJobID <id>, hasStartDate <start\_date>, hasEndDate <end\_date>

: scheduledOn Server = <node\_num\_in\_rack> that is atLocation Rack = <rack-num> and hasNodeType (↑hasType) <nodetypes>,  
: usesNode <numnodes>, and usesCore <numcores>, and requiresSharedNode <sharednode>

: was submittedAtTime <submit>, and was startedAtTime <start>, and was endedAtTime <end>  
: hasState <state> and hasExitCode <exitcode>  
: hasReservationIdentifier <reservation> ~~and hasPartPrepaid <partprepaid>~~ (Not enough clarity for ‘partprepaid’, nor public documentation)