Computation of Service Availability Metrics in Gridview

Digamber Sonvane, Rajesh Kalmady, Phool Chand, Kislay Bhatt, Kumar Vaibhav
Computer Division, BARC, India

Gridview computes Site and Service Availability Metrics using SAM test results. Computed metrics include Status, Availability and Reliability for individual service instances, individual services for a site, overall site, aggregate of all Tier-1/0 sites and central services.

1. Metric Definitions

- **Status** of a service instance, service or a site is the state at a given point in time. For eg. UP, DOWN, SCHEDULED DOWN or UNKNOWN.

- **Availability** of a service instance, service or a site over a given period is defined as the fraction of time the same was UP during the given period. For example, if a service was 'UP' for 15 minutes during a particular hour, the hourly availability will be 0.25 (fraction) or 25 % (percentage) for that hour.

- **Reliability** of a service instance, service or a site over a given period is defined as the fraction of time the same was UP (Availability), divided by the scheduled availability (1 – Scheduled Downtime – Unknown Interval) for that period.

  Reliability = Availability / Scheduled Availability
  = Availability / (1- Scheduled Downtime – Unknown)
  where
  Availability - is fraction of the time the service was UP (Tests Passed)
  Scheduled Downtime - is fraction of the time the service was Scheduled Down
  Unknown Interval – is fraction of the time for which the Test Results were Unknown (that is, validity of earlier test result expired and new test result not arrived yet)

  Unscheduled Downtime= (1 – Availability – Scheduled Downtime
  – Unknown Interval)

  Reliability = Availability / (Availability + Unscheduled Downtime)

- **Example :** Consider status of a particular service for a day as UP for 12 Hours, Scheduled Down for 6 Hours and UNKNOWN for 6 Hours. This would result in an availability value 50 % and Reliability value 100 %. Reliability is not affected by Scheduled Downtime or Unknown Interval.

- **The above definition of Reliability was approved in LCG MB Meeting dated 13th Feb, 2007.**

2. New Algorithm for Computation of Service Availability Metrics

Currently Service Status and Availability is being computed by sampling the test results at the end of each hour assuming that tests are launched on an hourly interval. This assumption, which was true initially, may not be true anymore with SAM Framework being enhanced to add the flexibility of launching tests several times in an hour. In principle, a test may pass and fail several times in an hour. So with the discrete time scale with precision of an hour, it is not possible to represent several events in an hour and there is also a loss of precise information about the exact time of occurrence of the event. For this reason, a new algorithm, which computes service status and availability metrics on a continuous time scale, is developed. **This conforms to Recommendation 42 of EGEE-II Review Report about introducing further measures of robustness and reliability, which in particular recommends (a) introducing time series data as well as snapshot data and (b) providing more finely-grained data.** In fact, we started working on these measures even before the recommendation was made.

The new algorithm also computes Site and Service Reliability Metrics as per the reliability definition approved by LCG MB in the Meeting dated 13th Feb, 2007.

3. Differences between Old and New Algorithms

There are four major differences in service availability calculation using old (current) and new algorithm.

a) Service Status computation on Continuous time scale
b) Consideration of Scheduled Downtime
c) Handling of UNKNOWN status
d) Validity of Test Results

The implications of these changes are mentioned below.

a) **Service Status computation on Continuous Time Scale :** The currently deployed algorithm computes availability metrics based on the Service Status over a discrete time scale with a precision of one hour. The new algorithm uses Service Status over a continuous time scale, the most fine-grained representation. With discrete time scale, the status is computed at hour boundaries considering only the latest result for the hour. For example for status calculation for a particular service for the hour between 9 and 10, if the result at 9:05 is UP and result at 9:55 is DOWN, the service will be considered DOWN for the entire hour. Where as, for the status computation on continuous time scale, the service status is based on all the test results and is more accurate representation of the true state of the service. With discrete time scale, the availability for an hour is always either 0 or 1, we never get a precise value. Where as, the availability computed from service status over continuous time scale is more accurate.

b) **Consideration of Scheduled Downtime:** Scheduled downtimes of services and sites, which are published in GOCDB are considered while computing the service status and availability in the new algorithm. In current computation, scheduled downtime is not considered during service status or availability calculation. So if a service passed test results even when scheduled-down, that is added up to the availability of the service, which is not desirable as it leads to inaccurate value for the reliability metric. In the new method, the status of the service is explicitly marked as SCHEDULED DOWN and the test results are not considered during this interval, so they don't add up to the availability of the service. In fact, this is a pre-condition to compute reliability as per the approved definition. Also if one of the critical services of the site is scheduled down the entire site is considered scheduled down. If scheduled downtime extends or shrinks, the site administrator can update it in GOCDB and updated interval will be considered, provided change is done before the interval is elapsed.

c) **Handling of UNKNOWN Status :** In the current method, availability computations are based merely on available test results, without explicit consideration of unknown test results, where as the new method takes all essential parameters into consideration. In the new method the status of all tests for all registered services is tracked continuously and the status is explicitly marked "UNKNOWN" wherever test results are not available and the validity of earlier test results is expired. It has impact on individual service instance availability as well as overall site availability.

For Individual Service Instance Availability, a VO marks a set of tests for a particular service as critical. A service should be considered UP and available only if all the critical tests are passed. The current (old) algorithm, which is driven just by available test results, marks the status of the Service as UP even if test result of only one of the many critical tests is available and UP, ignoring the status of other critical tests whose results are not known, whereas the new method marks the service status as UNKNOWN even if the status of all known critical tests is UP and one of the critical tests is UNKNOWN, when any of the known results is DOWN the status is marked as DOWN any way, basically it's an ANDing operation.

Moreover, the old algorithm computes the status of only those services for which at least one critical test is defined and the result of at least one critical test is available. This leads to unexpected behavior like a particular service instance suddenly disappearing from Gridview's monitoring page for the site, just because the test results are not available for a given duration. This make the possibility of the status of a particular service being monitored in Gridview a probabilistic phenomenon. Whereas the new method, tracks service instances, based on their registration in Information System, either in BDII or GOCDB. The set of VOs supported by each service instance is discovered from BDII and a relation between service instances and supported VOs is established. The new method deterministically monitors (computes/displays) the status of all service instances supporting the given VO, irrespective of whether critical tests are defined or test results available. The new method may say that the status of a particular service instance is unknown, but it at least recognizes the service and faithfully tracks its status.

We compute the overall status of the site from the status of individual service instances. We first compute the status of a service (here, by service, we mean a service type and not the individual instance), eg. CE, for a site by ORing the status of all redundant service instances, eg. instances of CE, for that site. For a particular service to be available at least one instance should be available. We classify services as site level services (CE, SE, SRM, siteBDII etc) and central services (RB, FTS, LFC etc). We consider it critical for the site, to provide all site level services advertised by the site. We compute the overall status of the site by ANDing the status of all site level services, so the overall status of a site should be UP only if the status of all site services is UP. The old method marks the overall status of a site as UP, even if status of only one of the many registered services is known and UP, ignoring other services for which the status is not known. The new method, marks the status of the site as UNKNOWN even if the status of all known services is UP and one of the services is UNKNOWN, when any of the known services is DOWN the status is marked as DOWN any way.

In summary, the old algorithm arrives on a conclusion based on incomplete data which can be sometimes misleading, giving the impression that everything is ok when the fact is not known. This may also have an adverse effect on the actual availability of the service as monitoring system might be hiding some of the service breakdowns rather than generating alerts. The new method generates the true status of the service, sincerely stating it as UNKNOWN whenever it doesn't have adequate data to establish the state of the service.

d) **Validity of Test Results :** The Current algorithm considers the validity of test result as 24 Hours for all test results. The new algorithm uses the validity period for the tests as defined by VOs. We are thinking of invalidating earlier test results after scheduled interruption of a service instance and marking the status as UNKNOWN until we get fresh results as there is high possibility of some parameter being altered during the interruption resulting in change of status.

In summary, the new algorithm computes the site and service availability metrics with more accuracy and resolves the ambiguities present in the current algorithm. The implementation of this new algorithm is ready, we are awaiting the management nod for deployment.

4. Metrics being computed

The following metrics are being computed:

1. Individual Service Instance Status, Availability and Reliability
2. Individual Service Status, Availability and Reliability
3. Individual Site Status, Availability and Reliability
4. Aggregate Tier-1/0 Availability

5. Central Services Status, Availability and Reliability

5. Computation of Service Status, Availability and Reliability Metrics

The status of service instances, services and sites (service instance status, service status and site status) is computed on continuous time scale for each supported VO using SAM test results. Availability metrics for service instances, services and sites are computed on hourly basis for each supported VO from respective status information. These hourly metrics are later used to generate metrics for different periodicities such as daily and monthly.

5.1 Computation of Continuous Service Instance Status: 'ServiceInstanceStatus'

This metric indicates status of a particular instance of a service, for example, a specific CE, over a period of time. This status is determined by using results of all the critical SAM tests that were run on the particular service instance. The criticality of SAM tests and validity period of the result is defined by each VO. Site Administrators can announce the Scheduled Downtimes for service instances in GOCDB. From all this input parameters, five possible status values are derived for the service instance.

- SCHEDULED DOWN: The service has been marked as under scheduled downtime. Test results are ignored in this case.
- DONTCARE: No critical test is defined for the service
- UP: All critical tests have passed
- DOWN: At least one critical test has failed
- UNKNOWN: No critical test is failed and result of at least one critical test is not available

5.2 Computation of Continuous Service Status: 'ServiceStatus'

This metric is computed by combining status of all instances of a particular service type (computed above), for example, all individual CEs in a site, at a given point in time. Here too, the same four status values are derived.

- UP: At least one instance of the service is UP
- DONTCARE: No critical test is defined for the service
- SCHEDULED DOWN: No service instance is UP or DONTCARE and at least one is 'SCHEDULED DOWN'
- DOWN: No instance is 'UP' or 'SCHEDULED DOWN' or 'DONTCARE' and at least one is DOWN
- UNKNOWN: All instances are of 'UNKNOWN' status.

Services are classified as site level services and central services. Site level services consist of CE, SE, SRM and sBDII. Services such as RB, LFC are considered as central services. For computing 'ServiceStatus' of site level services, only instances belonging to a particular site are considered, whereas for computing 'ServiceStatus' of central services, all instances of the service (irrespective of site) are considered.

5.3 Computation of Continuous Site Status: 'SiteStatus'

This metric is computed by combining status of all site level services (computed above) provided by the site (that is, registered by the site in information system, either GOCDB or BDII), viz, CE, SE, SRM and sBDII, at a given point in time. The following four status values are derived.

- DOWN: At least one service is DOWN.
- SCHEDULED DOWN: No service is DOWN and at least one is 'SCHEDULED DOWN'
- UNKNOWN: No service is DOWN or 'SCHEDULED DOWN' and at least one is 'UNKNOWN'
- UP: All services are UP or DONTCARE

5.4 Availability and Reliability Computation

- The Hourly Availability (status UP or DONTCARE), Scheduled Downtime and Unknown Interval for a service instance, a service and a site are computed on an hourly basis from the respective status information for that hour. For availability computation the DONTCARE status would be considered as good as UP.
- The Hourly Reliability of a service instance, a service and a site is computed on an hourly basis from the Availability, Scheduled Downtime and Unknown Interval for that hour.
- The Daily, Weekly and Monthly Availability, Scheduled Downtime and Unknown Interval figures are computed from the corresponding Hourly figures by averaging over the required time periods.
- The Daily, Weekly and Monthly Reliability figures are computed directly from the Availability, Scheduled Downtime and Unknown Interval figures over the corresponding periods and not by averaging the Hourly Reliability figures.
- The aggregate tier-1/0 availability figures are computed by averaging the availability figures for all tier-1/0 sites.