Velocity **Intelligent Resource Queuing**

A Just-In-Time reservation system for Velocity

Technical Benefits

- Build end-to-end automated workflows–from software development to integration testing to deployment–in a CI/CT model, with highly efficient resource utilization
- Use the 'Deferred' reservation option to allow *just-in-time* resource reservation– reserve and acquire the resource when it is needed and release immediately after use.
- Automatically queue topologies on reservation conflict; resolve and run on resource availability
- Get notified on the reservation status by registering callbacks via REST API

Challenge

Test campaigns often involve expensive resources such as high precision measuring instruments or high-end network equipment. These instruments are a scarce resource for an organization because it cannot afford to have multiple instruments, enough to satisfy the needs of development, testing, manufacturing and other teams. More importantly, resources are not used efficiently. Normally, all the required devices are 'locked' to a test run (topology) for that entire duration. As a result, other tests that need these resources are blocked, waiting to run. In a test run that takes a few minutes to several hours, these resources are actually required for a very short duration–a few seconds to few minutes. Idling away resources by confining them to a test run is very inefficient. Resources, scarce or not, expensive or not, need to be utilized more efficiently!

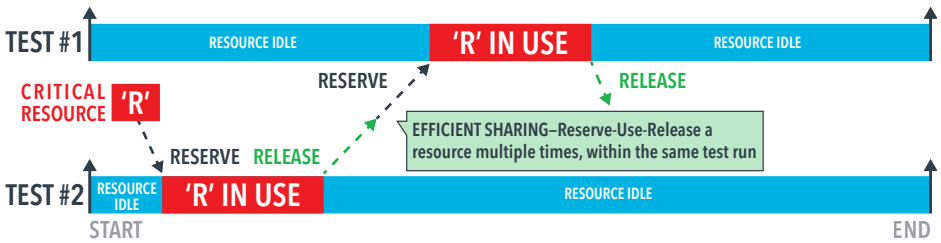
The Solution

Efficient reservation and resource sharing is the only way to meet the demands across teams, test campaigns and topologies.

Spirent Velocity 8.1 introduces Intelligent Resource Queuing feature that provides a highly efficient *just-in-time* resource reservation functionality. Using this feature, users can reserve resources at the time of need, and release them once done. This reserve-use-release sequence can be repeated many times, all in a single test run! This maximizes resource usage and promotes efficient resource sharing across teams, topologies and test campaigns. Resource contention and conflicts are handled seamlessly by queuing reservation requests in a FIFO order, with optional priority overrides.

Business Benefits

- Reduce CapEx significantly by completely removing resource idling and efficiently utilizing the resources
- Improve time-to-market by streamlining testing, service delivery and leveraging the new ability to run more test cases in parallel



Spirent **Velocity**

Intelligent Resource Queuing

Functionality

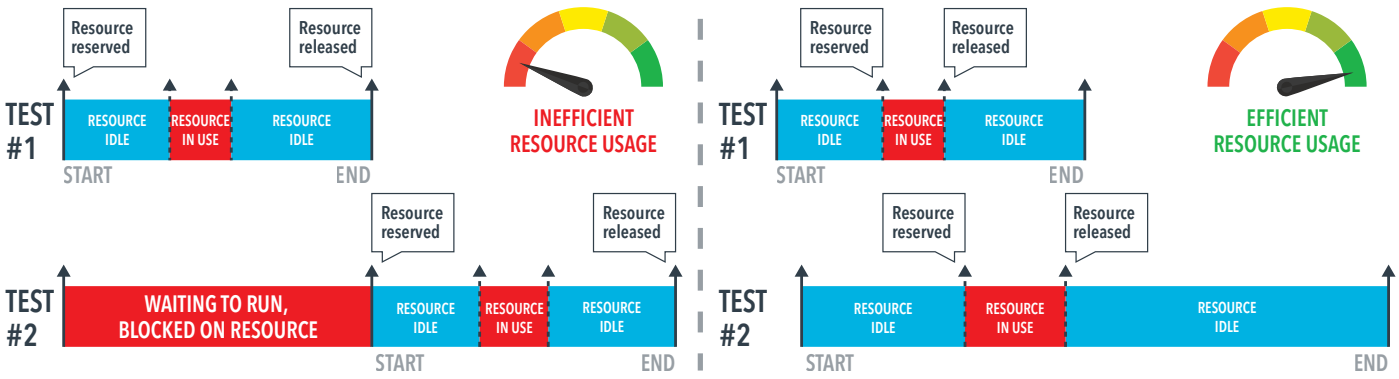
Velocity's Intelligent Resource Queuing allows sharing of limited resources among overlapping topology reservations. It provides two important functionalities:

- The ability to defer resources in a topology (not activated when a reservation becomes active), but manually activate them when actually required
- The ability to address resource conflicts by queueing reservations and automatically activate reservations as and when conflicts get resolved

This feature enables the creation and reservation of multiple topologies that require the same resource (e.g., expensive measuring equipment). That is, shared resources are required to be reserved only when they are actually needed and for the duration they are needed, and not for the entire duration of the topology reservation (test run).

Use Cases

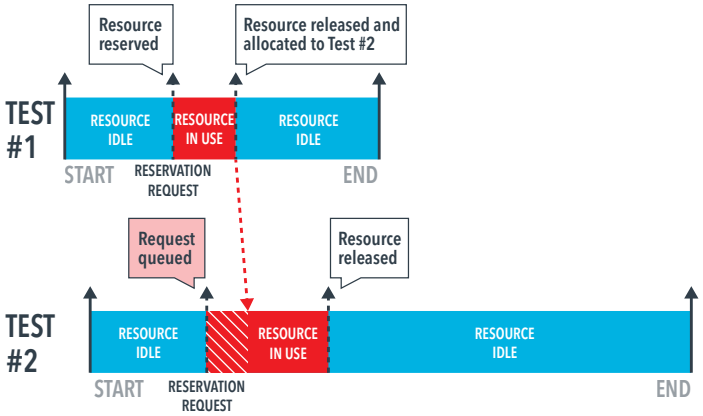
#1: Time Slicing: Share scarce resources across multiple testcases



BEFORE *AFTER (with Intelligent Resource Queuing)*

- Two users, User1 and User2 need to reserve the same 'scarce' resource, a 5G test equipment for their testcases. Both build their respective topologies and reserve this equipment as a 'deferred' reservation.
- The topologies are processed by the 'resolver', the reservation compiler inside Velocity. It prioritizes and assigns this scarce resource to the testcase that needs it earlier. It then informs User1 that the reservation requirements are satisfied, and that the testcase is ready to run. Optionally, it can start the execution.
- Once the resource is 'done' and no longer needed for that testcase, it is released and allocated to the other topology. User2's testcase is now ready to run. Note that User1's testcase can still be running when the 5G test equipment is released. Velocity's intelligent resource queuing feature time-slices to efficiently share resources to maximize their usage.
- Take-Aways:
  - Time slicing creates efficient resource sharing, independent of the testcase execution duration
  - Enables parallel testcase execution, even with resource scarcity and contention
  - Velocity can execute reserve-use-release sequence several times, often within the same testcase to achieve maximum resource usage

#2: Conflict Resolution: Addressing reservation conflicts by queuing



Conflict Resolution using Queuing

- A mobile equipment manufacturer has a scheduled software upgrade test at 2:00PM every Friday. 50 testcases need to run before and after the upgrade.
- Each testcase triggers a reservation request to Velocity, i.e. 50 requests in all.
- A pool of 20 radios are available for testing, even though the requirement demands 60 (some testcases need more than one radio)
- Scarcity of radios allows only some requests (less than 20) to activate immediately. The corresponding testcases (scripts) are informed of the activation status via callbacks.
- The remaining (30) requests are queued. As tests complete and radios become available, queued reservation requests are activated and their test scripts informed.
- Take-Aways:
  - Velocity queues corresponding topologies when resources are unavailable, and automatically executes testcases when resources become available.
  - Saves the user from manually resolving resource reservation conflicts and creating workarounds by building sub-optimal testing environments that lead to test escapes and support cases.

#3: Replicating a Customer Found Defect (CFD) for troubleshooting and RCA

- A network device manufacturer and a semiconductor manufacturer are collaborating to root-cause a serious network issue in a recently released network device. An in-depth analysis has led the engineering team to suspect a multi-service switch chip in the device. The teams need to replicate the issue before starting the debug session.
- To facilitate debugging, the chip vendor has mounted the chip on In-Circuit Emulator (ICE) pod and and exposed the relevant signals using mictor and JTAG cables, and ready to be hooked to the troubleshooting equipment. Logic analyzers and oscilloscopes, the equipment needed are in high demand. Multiple engineers on the hardware team need it in their projects.
- This issue appears after around 30 days of running continuously, that too inconsistently, making it hard to reproduce. It is dependent on the network traffic (a mix of voice, video and data). To help in troubleshooting, the device manufacturer has obtained traffic captures (PCAP files) from the customer, and is being replayed using tcpreplay. Spirent TrafficCenter (traffic generator) with a specific mix of traffic flows has also been added to mimic the production network.
- Spirent's Velocity's built-in network topology editor is used to build the network topology. It manages the entire test setup. Automation engineers have hooked the API callbacks to Velocity to pause (freeze) the DUT (network device with the chip on ICE), the Spirent TrafficCenter and other devices when certain pre-conditions that lead to this issue are met. When that event occurs, engineers will hook up the instruments, 'un-pause' the test and single-step through the event sequence that puts this chip into the 'bad' state.
- Using Velocity's intelligent resource queuing feature, the 'Resolver' processes the topology and reserves the resources accordingly. Because the logic analyzers and oscilloscopes are in high demand, they cannot be 'locked' to this issue replication effort. Hence, these are marked as 'deferred'. The resolver identifies when these scarce resources are needed and sets the right triggers to pause the test. Then the engineers to hook up these scarce resources, un-pause and single-step through the trouble-shooting process.
- By not tying-up the logic analyzers and oscilloscopes to this harness and idle them for most of the 30 day period, they can be used efficiently by the hardware team.
- Take-Away:
  - Critical resources that are in high demand can be very efficiently shared using Intelligent resource queuing.

References

- Spirent's Velocity Datasheet
- Spirent's Lab as a Service Platform, please visit www.spirent.com/solutions/lab-as-a-service

Contact Us

For more information, call your Spirent sales representative or visit us on the web at www.spirent.com/ContactSpirent.

www.spirent.com

Americas 1-800-SPIRENT
+1-800-774-7368 | sales@spirent.com

Europe and the Middle East
+44 (0) 1293 767979 | emeainfo@spirent.com

Asia and the Pacific
+86-10-8518-2539 | salesasia@spirent.com

Rev A | 12/19