Request concurrency

Modern browsers issue requests in parallel to download webpages faster. StresStimulus simulates concurrent requests by mimicking the browser's behavior based on the following test factors:

- Page breakdown information.
- Types of simulated browsers in the mix.
- Types of simulated network bandwidth in the mix.
- Server responses.

It does not use the information about parallel connections collected during the recording because this information is not entirely relevant to the emulated test conditions. Emulated browser types can be different from the browser type used for recording. Also, the server load during the test run is typically higher than the one during the recording.

The following behavior is emulated for every page and every VU independently:

1. Issue an initial request and wait for the response. During this time, all dependent requests are blocked.
2. Receive the primary response and handle dependent requests, which can be issued in parallel or sequentially.

    a. **Parallel dependent requests.** Dependent requests, representing webpage resources, such as images or style sheets, are typically sent in parallel, using multiple connections. The number of connections depends on the maximum number of available concurrent connections supported by the emulated browser. Each browser type has two limits: Max connections per host and Max connections per proxy. Both limits are enforced, so the number of connections per any single hosts and the number of total connections are independently limited. All other dependent requests, if any, are blocked (queued). As responses are being received, some of the queued dependent requests are being issued without violating the browser's connection limit constraint. Additional blocking rules are imposed to prevent issuing subsequent dependent requests until certain critical responses are received. These rules follow standard browsers specifications.
    b. **Sequential dependent requests.** Some dependent requests on a page must be issued sequentially, according to the page logic. For example, a video player on the page requests subsequent video fragments after receiving and processing previous fragments. The following are the situations that qualify dependent request to be issued sequentially:

        i. Some MIME types (e.g. Text/HTML, text, JSON, HTML, XML, SOAP, WCF the) are always requested sequentially.
        ii. Depending on the tested application, you can add additional MIME types whose requests must be issued only after receiving all previous responses. This will prevent dependent requests of these types from being requested in parallel with other dependent requests on a page. To do so, in Configure Test --> Advanced Options, in the property **MIME Types requested sequentially,** click the drop-down and enter additional MIME types. Separate multiple entries by ",". For example, enter "image, video," as shown below, to request all images and videos sequentially; or enter "video/mp4" to Request MP4 video sequentially. After modifying this property, launch test wizard and rerun autocorrelation.
        iii. To avoid sending the request out of order and preserve application integrity, StresStimulus has to issue requests sequentially that have certain conditions such as receiving **Set-Cookie** response header, receiving redirect response (301 and 302), or having an extractor.

        Info: By default, the Set-cookie response header is one of the qualifying factors to send requests sequentially. To suppress this factor and allow such requests to be sent in parallel, select Test Case, and in property grid set property, **Set-cookie header enforces sequential requests** to **No** (default is **Yes**). Requests still can be sent sequentially if other qualifying factors exist.

3. After the last dependent response was received, the current page is finished processing.
4. Optional think time delay is injected.
5. The VU is ready for processing the next page.

Info: Sent and received traffic cannot exceed emulated network type bandwidth limits. The traffic can be appropriately slowed down in order to achieve this.

You can change the default request concurrency by turning the page breakdown, as described in the next section.

Recorded concurrency

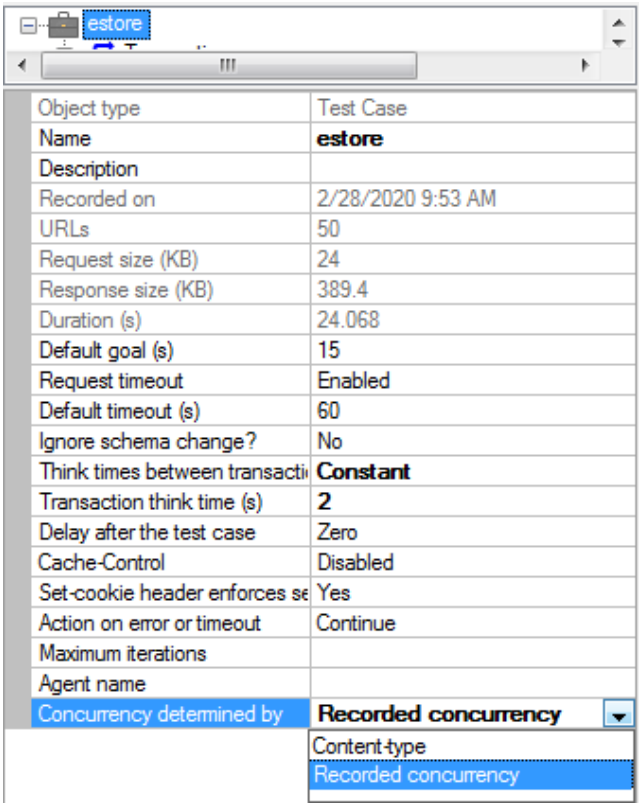
The above concurrency was determined by a request's MIME type (Content-type). Starting in v5.2, request concurrency can also be determined by recorded times.

When using recorded time concurrency, a request's recorded time is used to determine whether it is a parallel or sequential request.

- Requests A and B are sequential if, during recording, request B was issued **after** response A was received.
- Requests A and B are parallel if, during recording, request B was issued **before** response A was received.

So two requests issued in parallel during recording will be issued in parallel during the load test and vice-versa.

To change the concurrency mode of a test case, select the test case node in the test case tree, and set the **Concurrency determined by** property.



To view a test case's recorded waterfall, in the test case tree click the View Options button and select the Waterfall option