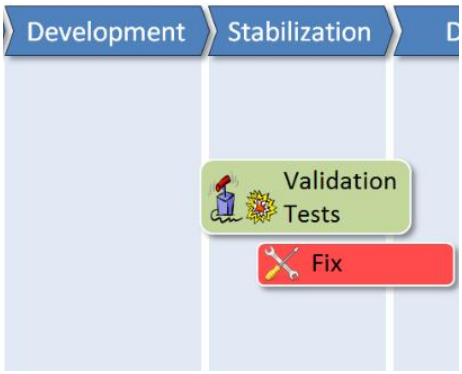
Performance Modeling Tests

While traditional performance load tests are the standard for *validating* the performance of a system, they are difficult and costly to run and often do not provide data that helps pinpoint and resolve problems. Additionally these tests are generally not feasible early in the project and are conducted too late to ensure performance issues can be addressed.

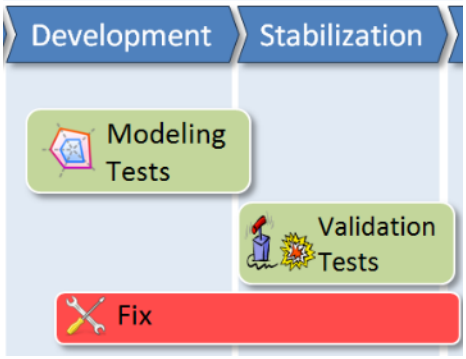
A testing approach called *modeling* can help to identify and resolve most key performance issues early in the project. In addition, these tests can also help to validate the infrastructure sizing prior to completing a successful validation test.

Validation Tests Only



Fixes start in Stabilization

Modeling & Validation Tests



Fixes start in Development

Comparing Validation Test and Modeling Tests

A modeling test focuses on understanding the performance of a single part of the application and answers different questions.

	Traditional Validation Test	Performance Modeling Test
What questions are answered by the test?	• Can the system support this number of users with acceptable response time? • Will the system batches complete in time?	• What is limiting performance? • Will this component or flow meet requirements? • What system resources are used by this component or flow?
Who runs the test?	• Performance testing specialists	• Developers or performance specialists
When is the test first run?	• Stabilization	• Development

While appropriate techniques for modeling tests vary depending on the type of component being tested, a properly designed test always answers the following three questions:

What is limiting performance?

Something is always keeping the component or flow from going faster. Identifying this provides performance insights that can lead directly to issues resolution.

Will this component or flow meet requirements?

Each component needs to do its part; there can be no weak links. A single integration can cause significant UI delays causing a system to fail to meet its overall performance requirements. Component focused modeling tests identify these weak links early allowing time for resolution.

What system resources are used by this component or flow?

Even if a specific component or flow will meet its individual requirements, it will need to do this in an environment with limited resources. Performance modeling extends component performance testing by identifying the resource requirements of the component or flow.

Capacity Projections from Modeling Tests

While the resource usage identified in modeling tests is meaningful and useful, it should always be complemented with validation testing to confirm how the complex system will perform. It is usually safe to assume that a component or flow will consume *at least* the amount of system resource identified in the test. The application server CPU usage is the metric most reliably projected. Database CPU can be impacted by more subtle changes in query efficiency and should be interpreted more carefully.

The table below shows the modeling data obtained for a business flow and includes resource utilization projections:

System	Metric	200 Claim Test (Measured)	Per Claim Cost (Calculated)	Peak Hour* (Projected)
Application Server	Heap Consumed	830 mbytes	4.15 mbytes/	7.47 gbytes**
Application Server	CPU Used	172 cpu-secs	0.86 cpu-secs/	1548 cpu-secs
Database Server	CPU Used (cpu-secs)	32 cpu-secs	0.16 cpu-secs/	288 cpu-secs
Database Server	IO Read	42,000 kbytes	210 kybtes/	378 mybtes
Database Server	IO Write (kbytes)	372,000 kbytes	1860 kybtes/	3348 mybtes

* Assuming 1800 claims per hour during peak.
** This is not peak heap as garbage collection will keep recovering the heap