# Optimus oscar.up\_GetPortfolioDetails Stored Procedure DB performance impact Analysis

## Background

A new stored procedure has been written to enable portfolio details to be derived from a port id which is supplied as a parameter. This stored procedure will be called every time a new dispute is created in Numero.

To ensure that the introduction of the new Stored Procedure for Optimus did not have a detrimental impact of Database performance it was agreed that some testing would be performed to establish there was no significant db performance impact.

## Approach

A test approach was agreed that would involve the Call Report team running their existing Performance Test suite and during the test an expected peak load would be executed through the stored procedure. This would enable to the Call Report team to identify if the stored procedure calls had any significant impact on their performance timings.

The desired load was to be created by simulating dispute requests being sent to Numero from CK/Ginger. As Numero processes a dispute request a call is made to the stored procedure using the Member Port ID from the request as a parameter.

## Volumes

Analysis of current production volumes of dispute requests received by Bizagi established that their average peak load is approximately 200 disputes per hour. Once a year the max load experienced was 400+ disputes.

## Tooling

SOAP UI was chosen as the tool to send the create dispute requests to the Numero API using the desired profile as this was already used by the test team to simulate single calls to the Numero API and approach could be easily expanded upon to simulate the required load in the required profile.

## Database Caching Considerations

Discussions with the DBA’s established that caching would occur in the database, which meant that we could not simply use the same test dispute request repeated to generate the load as this would mean the same parameter would be sent to the stored procedure for each call which would not generate realistic load due to caching. Therefore the disputes would need to have a number of different member port id’s to generate realistic load. This was achieved by parameterising the member port id in the dispute xml and randomly selecting a member port id from the available id’s each time a request was sent to the Numero API.

## Test Profile

It was established that Ginger sends disputes on the hour every hour, which would then be processed sequentially by Numero. This effectively means that an hour’s worth of disputes are sent through each hour in bursts, rather than tricking through at a consistent rate over the hour.

The SOAP UI Test was setup to send through the expected hourly load rapidly, with an aim to process all of the requests in less than a ten minute window.

## Test Execution

Testing was coordinated by the Product Delivery Team. They started their Call Report performance tests and monitored for approximately 15 minutes as the tests ramped up and stabilised. They then requested that the Optimus test was started.

Tests were started and configured to execute 600 dispute requests to the Numero endpoint. This took a total of 7 minutes to execute.

The Call Report Performance test was left to finish the full run which took approximately 90 minutes.

Start and finish times of the SOAP UI tests were provided to the Product Delivery Team so that they could check whether there was any impact on the Call Report Performance Test during this time.

## Test Results

The Call Report team analysed their response times during the period that the Optimus load was being produced, as highlighted by the Red Box below, and they established that there was no impact on their response times.

