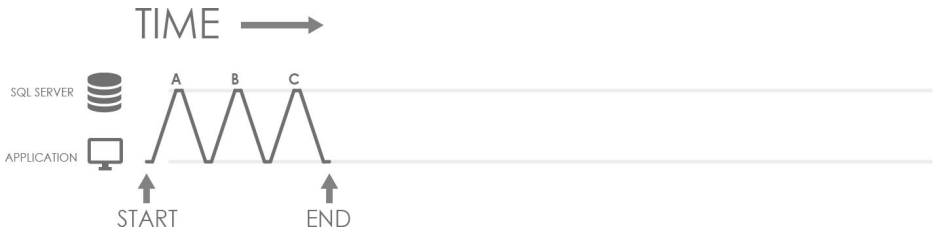
Data Accelerator

SQL applications over the internet/WAN

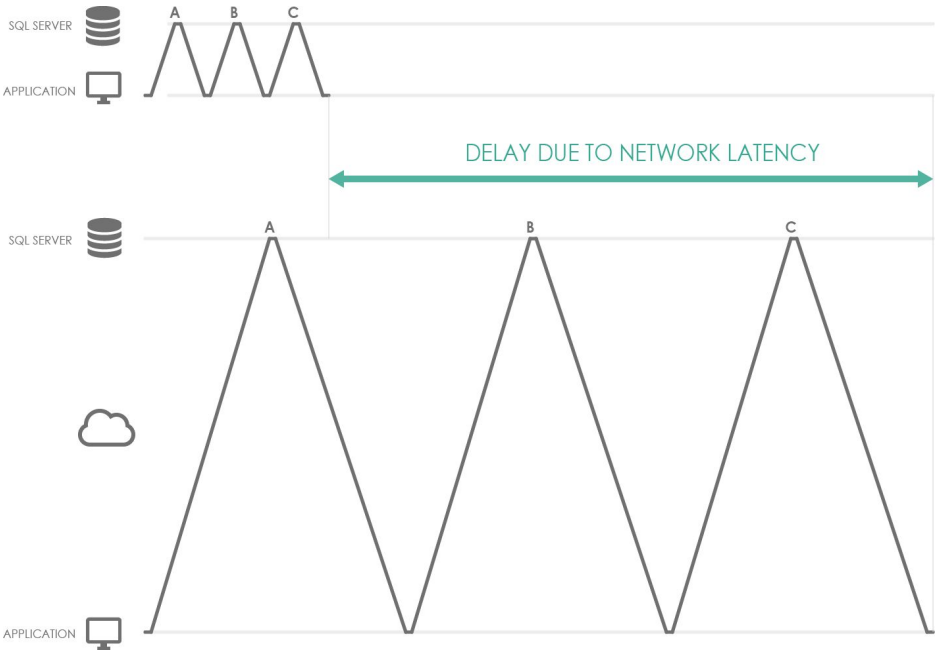
If your SQL application behaves fine on the local network but is slow over the internet or a WAN then there are two potential suspects: either your app is using all the available bandwidth or the app is “chatty”. In our experience, the problem is usually chattiness. While bandwidth issues can be easily sorted by upgrading your connection or kicking off the guy using torrents, chattiness is harder to solve.

What is chattiness?

A SQL application is said to be “chatty” if it makes multiple calls to the database for a single action by the user. For example opening the customer form in your app may seem like a single operation as far as a user is concerned, you just double-click on the customer and then the information appears, but if you look at a SQL profiler trace while that is happening you will probably see multiple queries hitting the database.



This diagram shows an application issuing three queries to the database, A, B and C, with time going to the right. Here A, B, and C represent the single operation for the application, “start” could be when the user double-clicks and “end” when the information is displayed. The point is that all 3 queries need to be executed in sequence before the app can respond to the user.

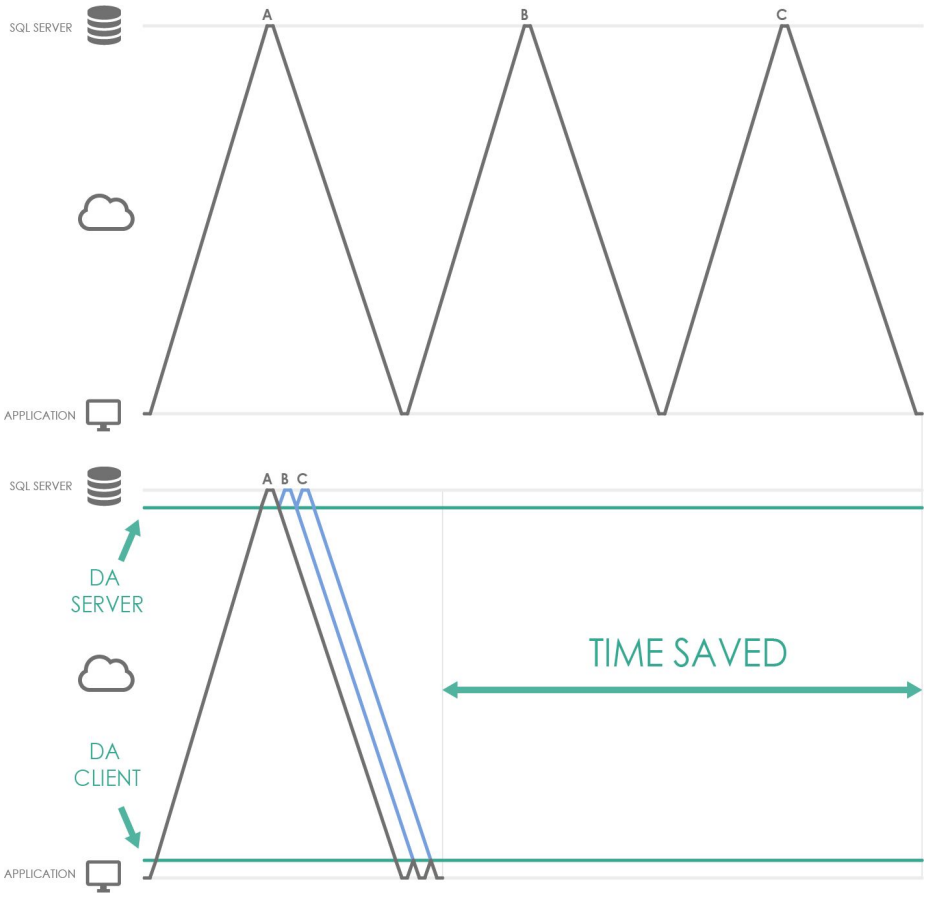


In this diagram you can see what happens when you try the same operation over a high latency network. The responsiveness of the operation is delayed by the network round-trips. The higher the network latency or the more queries you have, the bigger the problem.

Data Accelerator

Conversation Killer - How Data Accelerator combats chattiness

Data Accelerator uses machine learning to predict the queries your app will issue and prefetches them so the results are ready just before you need them.



The bottom half of the diagram shows how DA (Data Accelerator) speeds up the response time of the 3 queries. DA has a client and server component that together act as a proxy for the SQL traffic (using the TDS protocol). DA server monitors the requests that are issued by the application and makes a model of how the application issues queries through it's various workflows.

Once DA Server has seen a workflow the machine learning algorithms are able to make predictions of the exact queries that are coming up when the workflow next comes up. It predicts the TSQL statement or what stored procedures will be used, what the parameters will be, and what order the queries will be in.

In the diagram, Data Accelerator has noticed the sequence A, B, C before so that now when it sees query A again it can predict that B and then C will come next. Once it has received the response to query A it can issue query B. Likewise once it receives the response to query B it can issue query C. This way the queries are issued to the database as fast as any one connection can handle them, the only network round trips happening are the sub-millisecond ones between DA Server and SQL Server. The responses to B and C are streamed to DA Client which queues them up and providing the application does indeed request B then C it will be given the appropriate responses.

If the predictions are wrong and a different query from B follows A then the responses to both B and C are thrown away. The application therefore has to wait for the round trip to retrieve the response from the database, much as if DA was not there.

However discarding an incorrectly predicted query may not be enough, if the predicted query changes the state of the connection (for example with a Use statement) and the prediction is wrong, DA issues another query to restore the state of the connection to how it was before the predicted query was issued. Also writes (such as an Insert or a Delete) are not prefetched at all.

You might be thinking that these incorrect predictions increase the load on the database, and you'd be right but it is worth noting that incorrect predictions are quite rare when using DA. The algorithms are specialised for predicting deterministic processes (the code in the application) and actually stop making wrong predictions once it has learnt how the application works.

The algorithms learn very quickly. For workflows without variation such as when you first start the application, DA only needs to see the workflow once before it can achieve a high prefetch rate. Other workflows may have parameter value variations. For example displaying a Customer Information form will be a workflow that varies depending on exactly which customer it is displaying the information for. For workflows such as this, DA will generally provide some benefit the second time the workflow is run but will not be up to full speed until the DA has fully learnt how the various parameters are used. This generally takes DA server seeing the workflow 2 or 3 times but can be up to 6 times for very complicated workflows.

The model of the workflows is held in DA Server so only one user need experience an non-accelerated operation. Once that one user has completed the operation, all users will then have that operation accelerated. On a system with a large number of users it it becomes rare to experience a non-accelerated operation on the day that the learning starts, and typically non-existent thereafter.

Alternative products to DA?

There are various WAN Accelerators such as Riverbed and Silver Peak but none prefetch SQL Traffic. Without prefetching the only way to combat chattiness is with caching. The main problem with caching is how you expire the cache, and that can get very complicated. No SQL caching products in the market guarantee to maintain SQL's consistency model, they may return stale data. Adding caching to a system that was designed to work directly with SQL may introduce bugs. Riverbed's answer to this is to sell you man months of consultants to configure the cache expiry rules until the bugs seem to disappear. An alternative might be to rewrite your app to cope with the caching but we've already said that can be very expensive.

DA on the other hand is always getting the data fresh from the database so it is never more than a fraction of a seconds old. What's more the queries are never reordered so the latest response is always at least as up to date as the previous queries. This ensures that DA maintains SQL's consistency model. Everything SQL Server guarantees, DA also guarantees. This means it won't introduce bugs and the app does not need re-writing. The configuration is minimal (just tell DA server how to connect to the database) and so an accelerated connection can be set up minutes (in practice it's been taking us half an hour for a new app).

A free trial of DA is available here: http://dataaccelerator.com