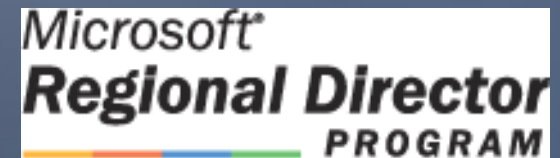


The case

A SQL Server 2005 Always On case?

Introducing...

- Class-A
 - Kennisprovider
 - Microsoft development
 - Training & Coaching
 - <http://www.class-a.nl>
- Anko Duizer
 - Trainer/ coach
 - <http://www.ankoduizer.nl>



Inhoud

- The Case
- Discuss
- Our solution

The case

The company

- Operating international
- 40 employees
 - Mostly sales and admin
 - IT department in Rotterdam & Paramaribo
- Founded in 2000

The application

- 150+ websites in different countries
- Search / Lead websites
 - Movers, Painters, Accountants, etc..
- All websites have the same structure using one database
- Classic ASP/ SQL Server 2000/ SQLXML
- The number of websites will grow in the near future...

The application

- Customers and suppliers
- Sales use a different admin tool to fill the database with contract information
- 500.000 leads a month, and growing...

The problem

- Instable solution – too much downtime of the websites
- The database is the hotspot, which is only available at one site
- It is not easy to deploy the current solution in other countries
- It is hard to hire developers with knowledge of SQLXML

The vision

- To develop a flexible application architecture that can be used to deploy many websites across different countries which are high available and scalable using the newest Microsoft technologies.
- Increase the number of leads

Design goals

- As generic a solution as possible
- Migrate the database to SQL Server 2005
- The current websites will only be changed when really necessary
- Increase the availability (99,9%)
- Increase the scalability
- It must be possible to add extra hardware without technical problems (at different locations)
- Replace SQLXML

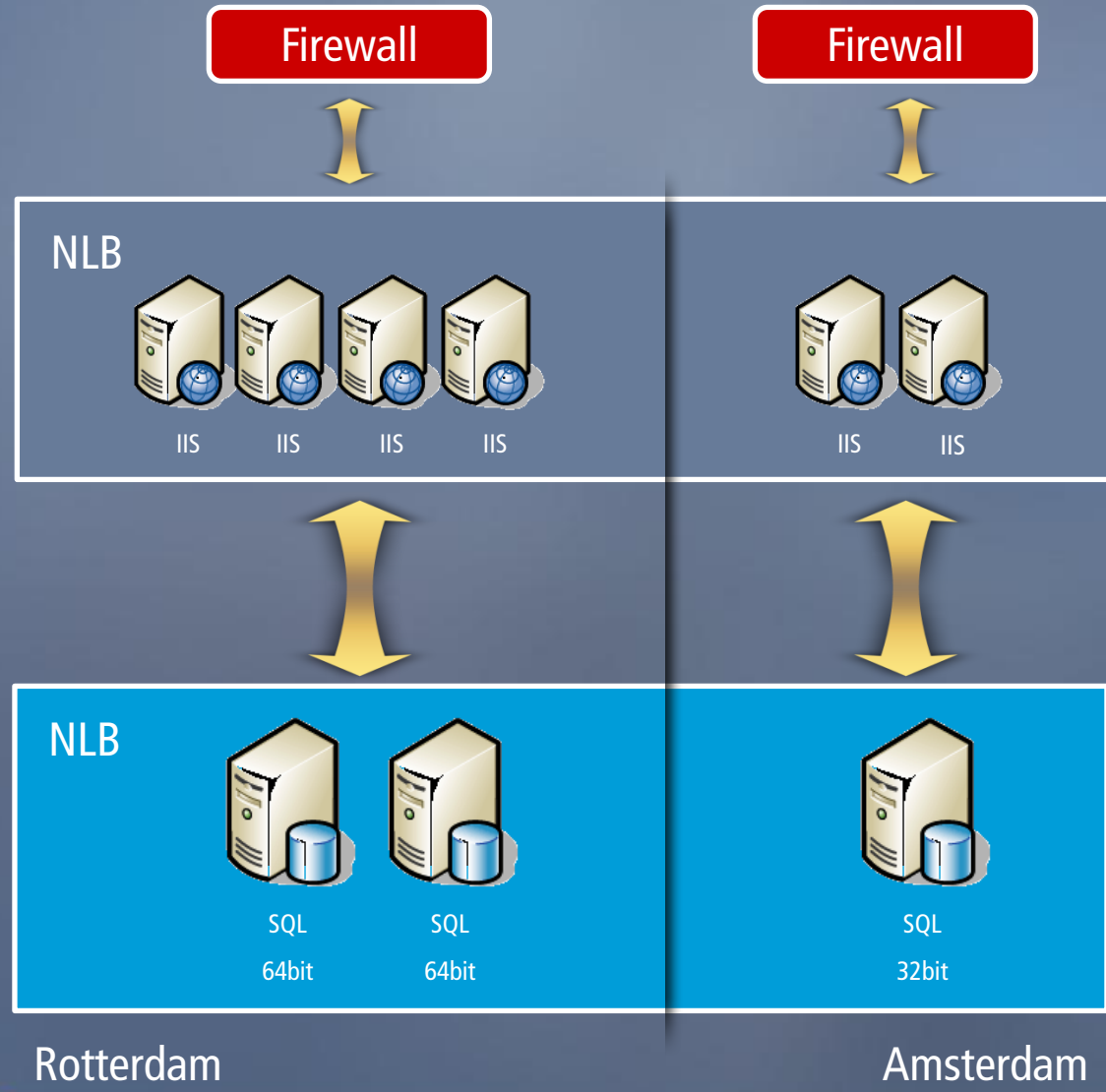
How to solve? Discussion...

Our solution

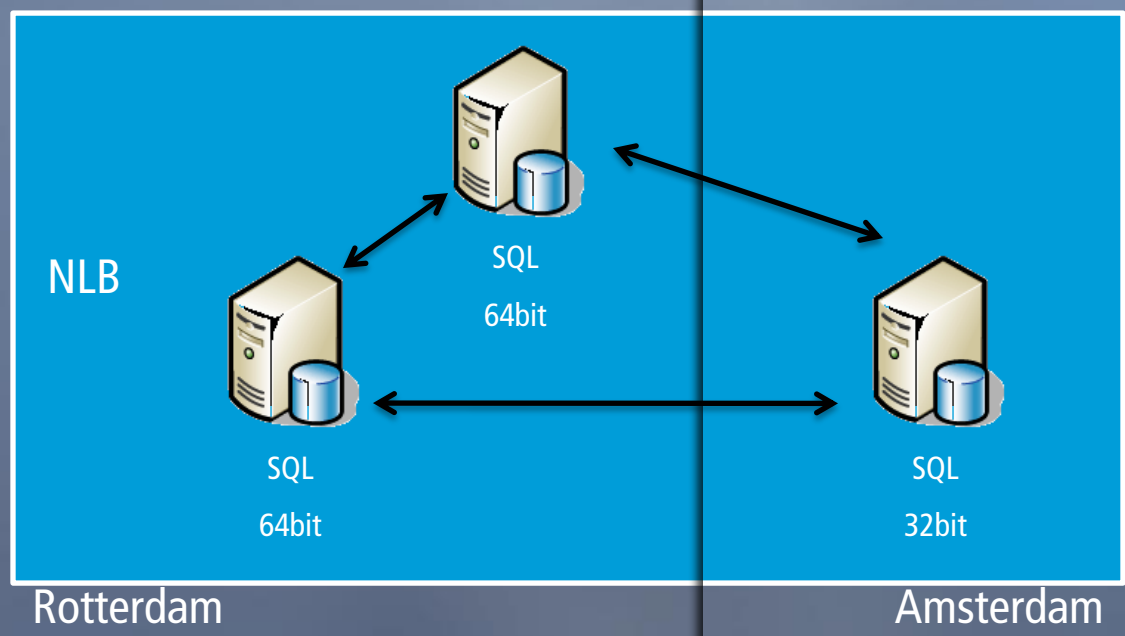
Global solution

- First upgrade the database architecture
- Replace SQLXML with ADO.NET
- Upgrade of the websites to ASP.NET is postponed
- Use NLB cluster technology for both the web and database cluster
- Use autonomous databases with peer-to-peer replication

Infrastructure / architecture



Peer-to-peer replication



Software design



websites XSL Transformation
ASP

XML formatting Data retrieval
ASP.NET 2.0

Stored Procedures Existing tables
SQL Server 2005

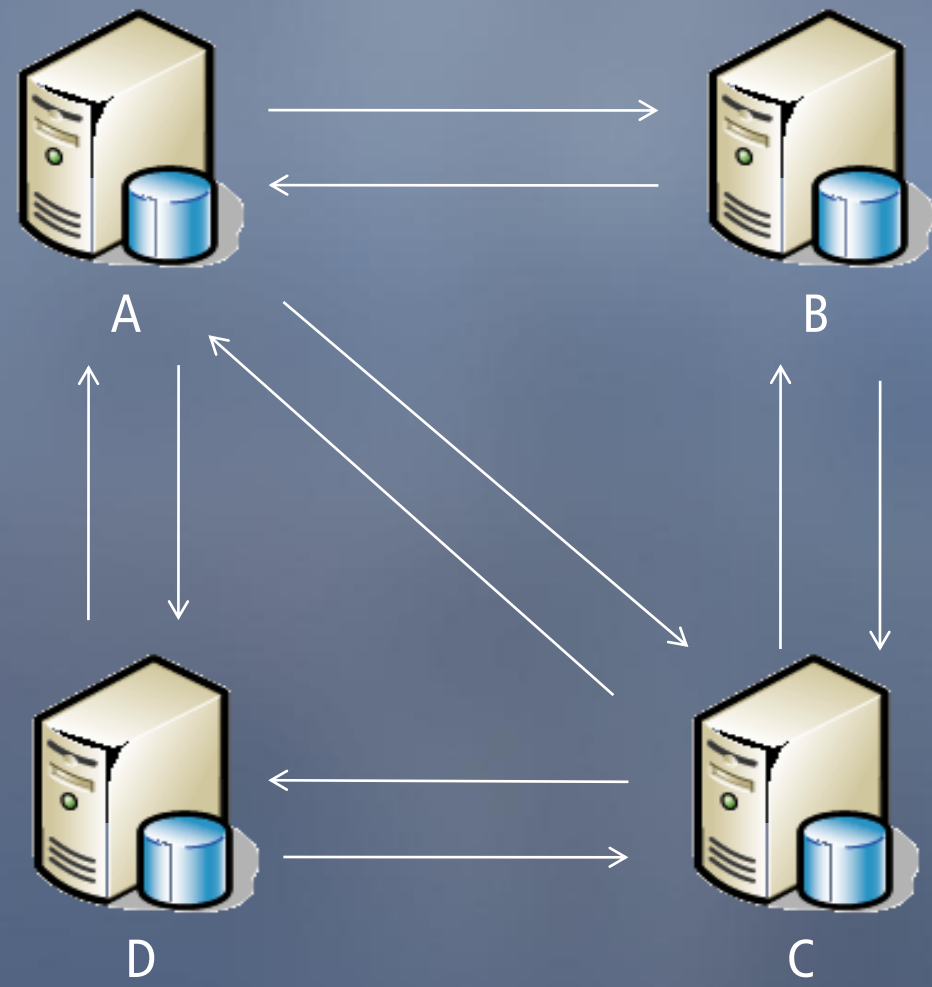
Why this solution?

- Combination of availability and scalability
- Relatively cheap
- Fits with the current application architecture
- Works at commodity hardware
- Microsoft.com uses the same architecture

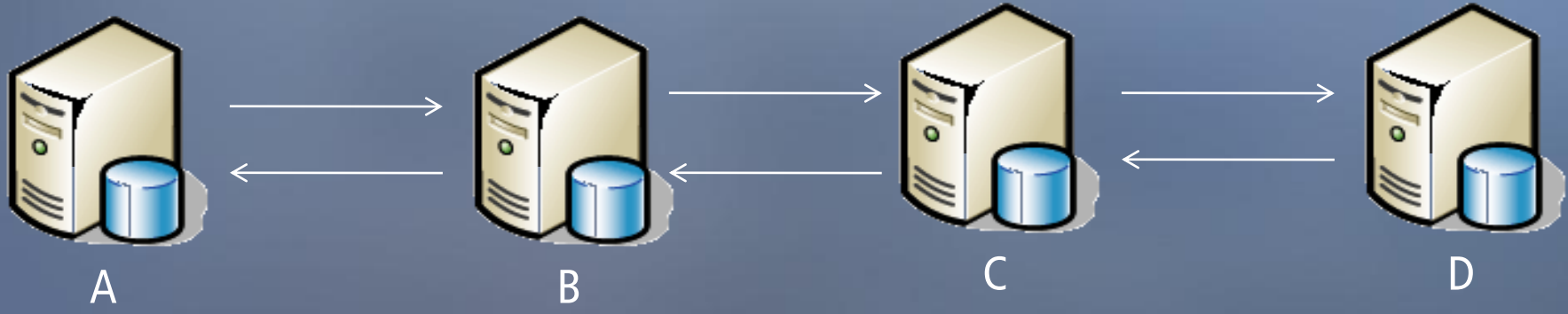
Issues

- NLB/ Peer-to-peer is not the same as failover
- Unique numbers
 - Ranges
 - GUIDs (which I hate...)
 - Combined with machine
- Replication can become complex...
- Error handling/ problem solving

Replication topology #1



Replication topology #2



Questions ?