This degree project has been carried out in collaboration with
Softronic AB
Advisors at Softronic AB: Tommy Paanola and Rikard Isoz

Monitoring an integration service in Windows Azure
Övervakning av en integrationstjänst i Windows Azure

MATHIAS KÄTTSTRÖM
TOBIAS TRUNEHAG

Degree Project in
Computer Engineering
First cycle, 15 Credits
Supervisor at KTH: Reine Bergström
Examiner: Ibrahim Orhan
School of Technology and Health
TRITA-STH 2013:29

KTH Royal Institute of Technology
School of Technology and Health
136 40 Handen, Sweden
http://www.kth.se/sth

Abstract

When selling a service to companies or individuals, it's important to deliver what you sell.

Currently, Softronic provides companies with a platform that provides different types of services. These services create opportunities to efficiently develop government functions in accordance with the Government's action plan for e-governance. These services are hosted by using the cloud computing platform Microsoft Windows Azure.

The services' functionality is important for both Softronic and their customers' businesses. To know early whether there is a problem or if there will be problems with the services, Softronic wanted to monitor one of these services with the help of an external monitoring service.

The aim of this thesis was to compare three different systems for monitoring cloud-based services, implement the most suitable one and develop an application for Windows 8 where customers can see the status of their message channel inside the service. The monitoring systems tested were chosen and judged according to Softronics needs and monitoring requirements.

A monitoring service called New Relic was chosen for implementation. New Relic has support for most of the required features, and for the rest of the features, an application was developed. A Windows 8 application and a website were developed that show information about customers' message channels, and a web service was developed to supply both of them with the data they need.

Sammanfattning

När man säljer en tjänst till ett företag eller privatpersoner är det viktigt att man levererar det man lovar. I dagsläget tillhandahåller Softronic en plattform med olika tjänster. Dessa tjänster hjälper till att skapa möjligheter för att lättare kunna utveckla myndigheters verksamhet enligt regeringens handlingsplan för e-förvaltning. Dessa tjänser är driftsatta i Microsofts molnplattform, Windows Azure.

Att dessa tjänser fungerar är kritiskt för både Softronic och deras kunder. För att kunna, i ett tidigt stadie, få reda på om det är problem med tjänsterna eller om det kommer bli problem med tjänsterna, ville Softronic övervaka en av dessa tjänster med en redan befintlig monitoreringstjänst.

Målet med detta examensarbete var att jämföra tre olika system för övervakning av moln-baserade tjänster för att slutligen implementera den bäst passande av de tre samt utveckla en applikation för Windows 8 där kunden kan se status på sin egen meddelandekanal. I arbetet har de valda applikationerna testats och bedömts efter Softronics behov och krav på övervakning.

En monitoreringstjänst som heter New Relic valdes för implementation. New Relic har stöd för de flesta av de nödvändiga funktionerna, och för de resterande funktionerna utvecklades en applikation. En Windows 8-applikation och en websida utvecklades för att visualisera kunders meddelandekanaler, och en webbtjänst utvecklades för att förse båda dessa med den information de behöver.

Preface

This report covers our degree project in computer engineering at KTH (Royal Institute of Technology), carried out in cooperation with Softronic Cloud Partner.

The content of this report requires basic knowledge of computer engineering.

We would like to thank Softronic, the Cloud Partner Team and both our supervisors at Softronic, Tommy Paanola and Rikard Izos.

We would also like to thank Reine Bergström, our supervisor at KTH and Johnny Panrike for letting us use his name(without his knowledge) as the working title of one of our applications.

Table of Contents

1. Introduction

This section will introduce the reader to the basics of the thesis.

1.1 Background

In the department of the company where this thesis was made, developers use Microsoft Windows Azure as their primary platform. Nearly all applications developed are cloud services and cloud applications. To be able to discover application bottlenecks and application errors in an early stage, the basic monitoring included in the management interface of Microsoft Azure is not enough. To resolve this need for monitoring, existing monitoring services/systems need to be researched and tested to confirm that all monitoring needs are met.

This thesis aims to give the reader basic knowledge about cloud computing and monitoring services in the cloud but also discuss pros and cons with third party monitoring services.

1.2 Goals

The goals for this degree project are mentioned below:

- Evaluate and implement a monitoring solution for monitoring services/applications in Microsoft Windows Azure. The service needs to support the following:
  - Custom configuration
  - Monitoring services in Windows Azure
  - Notify users (by email and/or SMS) upon errors
  - Notify users when a message in a Windows Azure Service Bus is deferred (not delivered).
  - Notify users when a Windows event is posted to an Azure Storage table.
  - Monitoring if a service is offline/online
  - Check workload on a host or instance
  - Measure the flow time throughout the application

- Develop two applications for visualizing a customer's service bus subscription(message channel);
  - A website that shows current and historical information.
  - A Windows 8 Store Application that opens the website in a browser and shows current information in its tile(icon).

1.3 Delimitations

It was decided in the startup phase of the project that the following limitations should be put in place:

- Only the service EF1INT should be monitored.

- The end service should not be monitored.

1.4 Working Conditions

The report should be ready and delivered 22/5-2013. A presentation should be held 12/6-2013. The project was developed using Scrum with two weeks iterations with a sprint demo ending each sprint.

1.5 Solutions

To propose a monitoring solution that works in Cloud Partners environment out-of-the-box with no knowledge of the current code, some research on both EF1INT as well as monitoring solutions has to be done.

When selecting candidates for this, we have to find out if they follow the criteria for what is needed in the monitoring solution. If most criteria are fulfilled, it is eligible for evaluation. The three with the most criteria fulfilled will then get evaluated. The one with a good fit that also gives a good impression shall be implemented in a simplified version of the production environment of EF1INT.

For the development of the Windows Store application, a study around the new Windows Store App system and how Modern UI works is necessary.

The project is carried out mainly on site at the company's office in Stockholm. Daily scrums with the project manager for project status and day planning will be held but also at the end of each sprint, a sprint demo will be held for other colleagues at Cloud Partner. After each sprint ends, a planning meeting and sprint retrospect will be held.

Version control is provided by a Team Foundation 2012 server in the cloud by Softronic.

2. Background

Softronic AB is a company that provides IT and management services, started 1984. The company has continued to grow since and now has about 500 employees. With a wide clientele knowledge, Softronic AB develops applications for a wide range of systems.

2.1 The EF1 platform

EF1 is a platform developed by the Softronic department Cloud Partner. With the use of EF1, Softronic provides services that create opportunities for an efficient way to develop the governments functions in accordance with the Government's action plan for e-governance.

EF1's range of services are divided into three major groups; Contact support services, Business support services and Infrastructure support services.[1]

2.1.1 Contact support services

These services provide the ability to establish user-friendly and accessible e-services for the citizens/businesses/organizations. The services streamline information gathering and dialogue between authority and the counterpart in a coherent manner.

Currently four different contact support services exist:

- EF1 - Mina sidor - A service for individuals/companies/organizations to access information about an agency's e-services, make requests, track cases and communicate with the authority.
- EF1 - Elektronisk datafångst - A service that provides the ability to identify users and manage signatures. The service also offers web forms that receive, complete, quality control, structure and deliver electronic input.
- EF1 - Informationstjänster - A service that provides possibility for the authority to provide customized information to customers about its externally orientated activities.
- EF1 - Söktjänster - A service that provides a powerful search engine that helps both internal and external users with keywords, suggestions, spelling and synonym management etc.

2.1.2 Business support services

The business support services are intended to contribute to an effective e-governance.

Currently four different contact support services exist:

- EF1 - Diarieföring och ärendehantering - simplifies, structures and automates the administrative work that is related to the exercise of authority and ensures that all legal requirements are met but also that defined workflows and processes are followed.
- EF1 - Arkiv - A service for preserving digital information over time to ensure that the storage is done and the information is searchable and traceable.
- EF1 - Portal - A powerful portal for creating and maintaining business support services and contact support services. It also provides the functionality to publish content and services on both external and internal websites.
- EF1 - Processtyrning - Provides the business with a powerful yet easy to use tool for defining, executing and measuring business processes.
- EF1 - Dokumenthantering - Provides support for effective management of a document's life cycle by controlling and supporting the creation, storage, retrieval, management, signing and distribution of electronic documents.

2.1.3 Infrastructure support services

- EF1 - Säker kommunikation SHS - Conveys information between the authority and other parties connected to the authorities SHS networks.
- EF1 - Gränssnitt mot kontroll av elektroniska legitimationer - The service includes verification of electronic IDs for both e-identification and e-signatures.
- EF1 - Mottagning/utskick och integration - Contains functions that are needed to create an effective and complete electronic process.

2.1.4 EF1INT

EF1INT is a part of the EF1 - Elektronisk datafångst in the group contact support services and integrates a Windows Azure Service Bus. EF1INT acts as a queue between sender and receiver. The sender could be a site, web form, program or another system and the receiver is usually the customers' servers (finance systems, government systems, email).

The service bus is basically a message queue with sub queues. Each sub queue can have any amount of subscribers. A subscriber can be a customer server or service. The subscribers are responsible for checking the sub queue for new messages. When a new message is found, a subscriber pulls the message from the queue and notifies the queue that the message was received. The queue then deletes its own copy. Thus, redundancy is achieved and it keeps the queue independent of the receiver. If the receiver is broken/down or anything else that prevents the receiver from getting messages, the messages are temporarily stored inside the queue for later.

The messages inside the queue are of a common type in the .NET library, BrokeredMessage. This message contains general attributes and a customizable one, the body. The body is the main load of the message and is XML based. The syntax of the body has to be known by both the sender and the receiver.

To make this service scalable and reliable as possible for both customers and the provider (Softronic), this service is hosted by the cloud platform Microsoft Windows Azure.

3. Theory

3.1 Cloud Computing

Cloud computing is the use of remotely located services. These services are delivered as a service over a network, typically the Internet. [2][3]

Providers of cloud computing often offer services according to three service models. The services are provided on-demand from service pools in the provider's datacenter.

3.1.1 IaaS - Infrastructure as a service

Infrastructure as a service is the most basic model. Basically it provides the user with a hypervisor such as XEN, VMware or KVM in the cloud. These hypervisors are often clustered to provide failover. Other resources that can be provided by IaaS are virtual networks, networks between virtual machines, file-based storage and software bundles.

3.1.2 PaaS - Platform as a service

In the PaaS model, the providers deliver a virtual machine, typically including operating system.

The most significant difference between PaaS and IaaS is that IaaS delivers an infrastructure. So for example with PaaS, no “local” network can be created between virtual machines.

3.1.3 SaaS - Software as a service

In the Saas model, the cloud providers install and operate application software in the cloud. The cloud users access the software from the cloud clients. Cloud users don’t manage anything underneath the software layer (operating system and infrastructure).

3.1.4 Advantages and disadvantages with Cloud Computing

The advantages of cloud computing are many.[4] The most primary advantages are:

- Cost efficient - Most cloud computing providers bill their customers by the actual amount of bandwidth used and calculations done by the datacenter.
- It's in the cloud - No need for a local server, no need for physical maintenance.
- Reachability - Today's cloud computing providers are so efficient with their redundancy, there is nearly no chance that the systems will be unreachable.
- Easy to access - In the cloud, the information is accessible from anywhere where there is an internet connection.
- Backup and Recovery - When in the cloud it is easier to create and store backups. Most cloud computing providers usually handle recovery information which makes the process simpler.

When having an application or system deployed to the cloud there is no need for a local server to run them anymore. While there are many advantages with cloud computing, it also has its disadvantages. For example:

- Technical Issues - Even though the data in the cloud can be accessed at any time and from anywhere, there are times when the system can have some serious dysfunctionality.
- Security - Security is a major issue in the cloud. When deploying applications to the cloud, one must have in mind that all sensitive information is now surrendered to the cloud computing provider. This could potentially put the company at great risk and can be a major factor when choosing cloud computing provider.

  Also, when storing information in the cloud, it could make the company vulnerable to external attacks while retrieving and sending data to and from the application.

3.2 Windows Azure

Windows Azure is a cloud platform created by Microsoft for hosting various different services with high availability and scalability. Azure was introduced in late 2008 and today the platform supports various different services that run in the cloud. The services can be created and managed from the Windows Azure management portal, which has an online interface that is easy to use.

3.2.1 As a Service

Through the Windows Azure Portal, Microsoft provides all three types of cloud service models.

The IaaS support for Windows Azure provides the user with the ability to create virtual networks and local networks between virtual machines.

At the platform level, Microsoft offers virtual machines with a Windows Server or Linux installation. Some OS images are provided by the portal for installation of a server, but you can also provide your own customized image with any distribution.

Microsoft also provides application hosting and website hosting. This comprises the SaaS offering for Azure.

For all Windows Azure services, a location has to be selected. This is due to the geographical limitations of network transfer. Windows Azure has major regions divided into sub-regions.

These regions are:

- Asia
    - East (Hong Kong)
    - Southeast (Singapore)

- Europe
    - North (Ireland)
    - West (Netherlands)

- United States
    - North Central (Illinois)
    - South Central (Texas)
    - East (Virginia)
    - West (California)

3.2.2 Storage

Azure has two options for storage, SQL Databases and something called Azure Storage.

Azure Storage is a collective name for table storage, queue storage and blob storage.They can contain up to 100 terrabytes of data each. To make data redundant, Microsoft implements a method they call GRS or Geo Redundant Storage. With GRS, all data is replicated to another location within the same region. This enables failover in case there is a major failure at the primary site. For local redundancy all data is replicated three times within the same data center.[5]

Table Storage is based on NoSQL key-value technology and is therefore perfect for storing large amounts of data that is loosely structured.[6] Queue Storage is used for storing small messages as a queue, to use for example as a workload-queue for an application, and Blob Storage is used for storing unstructured data, for example an image.

3.2.3 Service bus

The Windows Azure Service Bus is a technology for bidirectional message communication between service-oriented applications. A service bus has a main queue and the main queue has sub queues. The subqueues and the main queue can have subscribers. When a message is sent through the queue, a subscriber will be able to read the message. The client has to poll the message from the queue, but it's still very efficient.

The Windows Azure Service Bus has two types of messaging models, brokered messaging and relayed messaging. The brokered model uses a broker or intermediary in the message channel to provide durable storage messages passing through. An advantage with the brokered messaging approach is that the service does not need to be online. The client can still send messages to the queue. Another advantage with brokered messaging is that when a burst of messages reach the service bus, it can store the messages until all of them are processed. Relayed messaging, as opposed to brokered messaging, exposes an endpoint for the clients to connect to. When many clients are connected at the same time, it can result in reduced performance and timeouts. Due to the exposed endpoint, scalability is harder to achieve than when using brokered messaging.[7]

3.2.4 Monitoring

Azure has some native monitoring features which can be divided into four areas[8][9];

- Application status - In the management portal, users can see whether their applications' instances are running, restarting etc. A REST API is also provided to check this data programmatically.
- Diagnostics - Using an API available in Azure applications, diagnostics collection can be activated. This includes performance counters, Windows event log collection, crash dumps and custom logs created by the application. These logs are stored locally and sent to an Azure Storage account periodically.
- Storage Analytics - Diagnostics on Azure Storage accounts use predefined metrics in the Azure portal. There are currently two levels of metric collection, minimal and verbose. Minimal gives you minimal information, and verbose more detailed information. Verbose metrics collects data on a deeper level and enables closer analysis of problems that occur during runtime.
- Database Dynamic Management Views - These show information about database server health, execution times etc.

3.3 Windows Communication Foundation

Windows Communication Foundation (WCF) is a framework used to help with development of service-oriented applications, where data can be sent asynchronously between two endpoints. A data contract is set up, specifying how to communicate and what kind of data can be requested. The data contract is then supplied to the clients that use the service.[10]

3.4 Windows event log

Since Windows NT, all errors and events happening in Windows are logged. Every event has an event level, warning, error or information, an event id, an event source and a timestamp. Microsoft categorizes events in four classes.

These classes are:

- Application - Basic application
- Security - Mostly account information such as login and logout events.
- Setup - Package updates.
- System - Everything that is system critical i.e DNS, IIS and other services.

4. Monitoring software

This section describes criteria for the monitoring service/system as well as results from testing and evaluation of monitoring service/software candidates.

4.0.1 Risks with monitoring

Most risks when monitoring a system or a service involves the fact that the monitoring will affect the performance of the system being monitored. This is very important when working with systems that can't do asynchronous operations i.e. databases and one-threaded applications. This issue will make the system/service unresponsive when the monitoring system reads the data, which leads to a performance issue for the service.

Another risk that is relevant when using an external service as monitoring software is that sensitive information could be stolen by the service provider.

4.0.2 Requirements on the monitoring software

The guidelines for what needed to be monitored were determined by the EF1INT implementation. Both Azure web roles and worker roles needed monitoring because they are very critical in the workflow of EF1INT. The intermediate service bus also needed monitoring. To find products that supported all features needed was very hard. There are a lot of monitoring products on the market, but only a handful of those have support for Windows Azure and even fewer support the features needed.

The requirements from Softronic were:

- The system is configurable
- The system can monitor a services in Windows Azure
- The system notifies users (by email and/or SMS) upon errors
- The system can continuously monitor if a service is offline/online
- The system is able to check workload on a host or instance
- The system is able to measure data in a Windows Azure Service Bus
- The system can observe windows event logs in Azure Storage
- The system is able to measure the flow time throughout the application

4.0.3 Candidates

To find appropriate candidates, research of monitoring software was done. In the first part of the research, ten candidates were looked at. Due to the time limit, there was no chance of doing a thorough evaluation of all ten candidates. According to the information given from the sites of each candidate, not all had the functionality needed. Therefore, three candidates with most of the functionality were selected for evaluation; New Relic, System Center Operations Manager and Foglight.

4.1 Differences between candidates

All three of the candidates have some basic support. All of them are able to send email/notifications to notify users. The triggers for these actions are individual for each service/system.

Both New Relic and Foglight is web based and hosted by them. SCOM, however, needs to reside on its own server, either locally or in the cloud.

4.1.1 Host monitoring

Two of the candidates support this, SCOM and New Relic. Both candidates need an agent installed to support this feature.

Due to the fact that System Center Operations Manager has to be installed on a Windows Machine it has a lot more features than the web based New Relic.[11]

As opposed to New Relic, SCOM also supports agentless monitoring. This means that when the SCOM server is setup to support this feature, all error reports have to be sent to the SCOM server instead of the Microsoft error collector service.

4.1.2 Database/storage monitoring

Database and storage monitoring is also only supported by two candidates, Foglight and SCOM. Foglight only supports this if the database resides in the cloud.

4.1.3 Azure web application monitoring

For both Foglight and New Relic, this is a feature that comes out-of-the-box. SCOM has a management pack that can be installed to provide this.

4.1.4 Service Bus monitoring

None of the three applications have any native support for this.

4.2 New Relic

New Relic is a web based monitoring system for web applications, hosts and mobile web services developed by the San Francisco based company New Relic. It helps customers to discover bottlenecks in the web application workflow. All features are available through a web interface located on New Relic servers, so nothing has to be installed on a server to make it work.

4.2.1 Features

New Relic provides an easy to use interface with a lot of statistics and other important information.

Some of the features supported by New Relic[12]:

- Email notifications
- Infrastructure monitoring
- Server monitoring
- Application monitoring
- Weekly performance reports
- Custom Dashboards
- Real-user monitoring

4.2.2 Installation

When installing New Relic on applications hosted in Windows Azure, an installation file has to be included in the Visual Studio project. A cmd-file also has to be included, which runs the installation through the use of msiexec.exe. The cmd-file also contains information about the installation, such as a key that identifies what New Relic account to send information to and if it should be a full installation or not. When the project is published to Azure, the installation is done through the use of a start-up script that runs the cmd-file.

4.2.3 Overall impression

The overall impression we get after using New Relic for a couple of days is that it is really solid.

It displays relevant data in beautiful graphs, and with custom dashboards, the user can decide exactly what data will be shown. It is also very easy begin to monitor applications. To add an application to the monitoring pool, installing a NuGet package to the Visual Studio project will get the monitoring up within minutes.

The documentation however, is lacking in many areas, detailed information is hard to find and in some areas non-existent.

4.3 Foglight

Foglight is a monitoring service provided by Dell. Quest was the original developer of this product but they were later bought by Dell, who has continued to develop the product. Foglight has an easy to use interface and is very easy to get started with. Foglight supports monitoring of different platforms, not only Windows Azure. Databases, middleware, infrastructure and virtual desktop diagnostics monitoring is also supported.

4.3.1 Features

Some of the features supported by Foglight[13]:

- Email notifications
- Azure SQL Database monitoring
- Application availability monitoring
- Traffic location mapping
- Response times of an application from different locations
- Azure platform status (If Azure is up or down)
- Application Service Quality

4.3.2 Installation

Foglight is very easy to set up with Azure. To “install” Foglight, a certificate needs to be uploaded to the Azure Management Portal. When the certificate is uploaded, Foglight can read data from Azure about the applications, servers and databases.

4.2.3 Overall impression

It is really easy to install. Just upload the certificate that is received from Foglight to Windows Azure. And it works. It feels like it is just basic monitoring and some graphs. With Foglight there is no room for customization. If there’s a need to monitor anything that is not included with Foglight, it cannot be monitored.

4.4 System Center Operations Manager

System Center Operations Manager(SCOM) is a part of the System Center suite. System Center is a platform for cloud, infrastructure and virtualization management. System Center was mostly tested because it is used in-house.

System Center can be extended with[14]:

- Operations Manager - Provides flexible infrastructure monitoring, both private and public.
- App Controller - Provides a self-service experience that helps to easily configure, deploy and manage virtual machines and services in public and/or private clouds
- Data Protection Manager - Enables disk-based and tape-based data protection and recovery for server-based products, desktop products and bare metal (non-virtualized machines).
- Orchestrator - Automates creation, monitoring and deployment of resources in the infrastructure environment.
- Service Manager - Provides an integrated platform for automating and adapting best practices in the infrastructure. Also provides incident, problem, and asset management.
- Virtual Machine Manager - Provides management for virtualized data centers, virtualization hosts and networking for private clouds.

4.4.1 Features

Some of the features supported by SCOM [15][16]:

- Notifies users by e-mail, SMS and Instant Messages.
- Custom Management Packs can be installed and developed by users to provide monitoring support for custom applications (Azure)
- Monitors the Windows Event log.
- Is very flexible and scalable.
- Leverages Active Directory for user and group management.
- Is able to monitor non-Windows systems.
- Can create custom reports for users and applications.

4.4.2 Installation

System Center needs a lot of prerequisites to run in an environment. Under the evaluation part of this project, the installation of SCOM was the most time demanding task.

For the installation SCOM needs:

- Windows Server Operating system. Under the evaluation, Microsoft Windows Server 2012 was running on a virtual machine in Windows Azure.
- A backing Active Directory. Best practice is to not run Active Directory on the same server that runs SCOM.
- A backing SQL Server (Either SQL Server 2008 R2 or SQL Server 2012). Best practice is to not run the SQL Server on the same server that runs SCOM.

Unfortunately, none of the best practices were applied during the evaluation part of this project due to the short time available.[17][18]

4.4.3 Overall impression

SCOM was the candidate that was most time consuming. Mostly because a Windows server had to be set up, an active directory created, SQL database installed and other required prerequisites had to be installed. If the best practices had been applied a small corporate network had to be set up. If all this had existed already, the installation of SCOM had been much easier.

When it finally was installed and running, the connectivity to Azure has to be made. Because of non-native support for Windows Azure, a management pack needs to be installed. To actually connect SCOM to Azure, the developer certificate for Azure has to exist on both the Operations Manager server and Windows Azure. By following different guides and trying to accomplish this, no connection to Azure was established. SCOM is therefore untested when it comes to monitoring.

4.5 Software choice

Because of the impressions, basic abilities and customization abilities New Relic has, it was chosen to implement in a test environment. The test environment is a simplified copy of the production environment of EF1INT.

5. New Relic

This chapter briefly explains how the features of New Relic work and how they can be used.

5.1 Agent

The New Relic agent for .NET is the service you install on the server running the applications you want monitored. The agent starts with IIS and runs in the same process as the applications that are monitored, so there is no service or process with the only purpose of running the agent. The agent injects bytecode into the methods of the application to be able to collect metrics, uncaught exceptions and detailed information about slow transactions. The metrics are then reported to the New Relic service once every 60 seconds.[19]

5.2 Metrics

The New Relic agent collects performance metrics on predefined methods relevant to .NET applications. These metrics include execution times for database queries, page load times and the time it takes to receive a message from an external source. The methods to collect metrics from are defined in an xml-document, so a user can add their own.

5.2.1 Custom metrics

If there are metrics important to the application that are not already collected by the agent, custom metrics can be reported to the New Relic service in two different ways. If there are user defined methods doing calculations that may take a long time and the user wants to see how long a specific method takes to execute, the user can define those methods in an xml-document. Performance metrics on the specified method are then collected by the New Relic Agent and sent to the service. The other way is to use the New Relic Agent API to send key-value pairs to the service. Custom metrics reported using the New Relic Agent API can be anything, for example the current value of a variable or the return value of a method. The custom metrics can then be viewed separately in graphs/tables in the New Relic service, showing the method name or the specified key as the name of the metric.[20]

5.3 Web Transactions

A web transaction is any method that receives a message from an external source. Information about web transactions is recorded and reported to New Relic, individually if a slow transaction is noticed and also as an aggregate of transactions recorded over a minute. The information recorded includes what outside source sends the message, what methods are included in the transaction and how long each method takes to execute. A transaction is considered slow if the time it takes to send/receive is slower than a value defined by the user.[21]

5.3.1 Cross-application tracing

Cross-application tracing is a feature to trace transactions between applications running the New Relic agent. This enables users to see the methods invoked and the time they take in both the sending and the receiving application. To be able to do this, A New Relic ID is added to the http request to be able to recognize the message when it arrives at the destination application.[22]

5.3.2 Custom transactions

Not all web transactions are noticed, because not all transactions are known to New Relic. If a transaction isn't being traced, custom instrumentation can be made. Custom instrumentation is done by creating an xml-document specifying what methods are included in the transaction that should be traced.[23]

5.3.3 Key transactions

Transactions that are very important to an application can be set as key transactions in the New Relic service, meaning error rate and response time thresholds can be set for the specified transaction. When error rates or response times are too high, a notification can be sent to users of the service.[24]

5.4 Real user monitoring

Real user monitoring is a feature that shows actual response time for the end users, their location and the browser used. To record the response time, New Relic instructs the browser to start a JavaScript, collecting information from the time the user sends the request. When the page has finished loading, the script sends the information to the new Relic service.[25]

5.5 Error Collection

The New Relic agent automatically collects unhandled exceptions and reports them to the service. If an exception is handled, but the user wants the exception to be seen as an error in the New Relic service, the exception can be reported by using the New Relic agent API.

5.6 Alerts

New Relic has support for alerting users of high error rates and slow response times. Before an alert is sent, an event is created. If the event is open for three minutes, an alert notification is sent to the users in one or more of the following ways:

- E-mail
- Webhooks - JSON-encoded data sent to a user specified URL.
- External services - PagerDuty(sms), Campfire and HipChat.

For each application, thresholds for error rates and response times can be set in the New Relic service. For error rates, the threshold specifies a percentage of requests resulting in an error that should be tolerated. When a threshold is breached, a notification is sent to users that subscribe to alert notifications from the application/key transaction from which the alert originated.

5.7 Performance reports

Users can get weekly reports sent automatically with detailed information about the performance of your system. Reports can also be generated by a user to send to anyone on command.

5.8 API

New Relic has two different kinds of API's; the Agent API to help with reporting and the REST API to communicate with the New Relic service directly.

5.8.1 Agent API

The agent API has functionality for reporting custom metrics/errors and ignoring parts of the code for the automatic collection of metrics.

5.8.2 REST API

The REST API has functionality for sending and requesting data to/from the New Relic service. This includes changing or reading New Relic-specific settings and requesting metric or error rate data.

6. Result

This section provides information about products developed by us. These products were developed to satisfy the needs from Softronic.

6.1 Service Bus Visualization products

Two of the goals of this project were focused on visualizing the status of customers' Service Bus subscriptions (message channels). The first was to create a website that shows current and historical information. The second was to create a Windows Store application that shows current information in its tile and opens the website.

Since both of the applications were supposed to show some of the same information, a third application, the Data Provider, was developed to provide them both with the data needed. This could also be done by each application independently, but every reconnect to the Windows Azure Service Bus cost a lot of resources and some of the same calculations would be done by different applications. To avoid this, the Data Provider works as middleware. According to Microsoft, this is one of the best practices to retrieve data from a Windows Azure Service Bus.

How they work together is described by *Figure 1* below. The Data Provider gets information on the current status of the message channel directly from the Service Bus. It's also connected to an Azure Storage account that has a table with the historical information needed. The website and the Windows Store Application get all the data they need from the Data Provider.

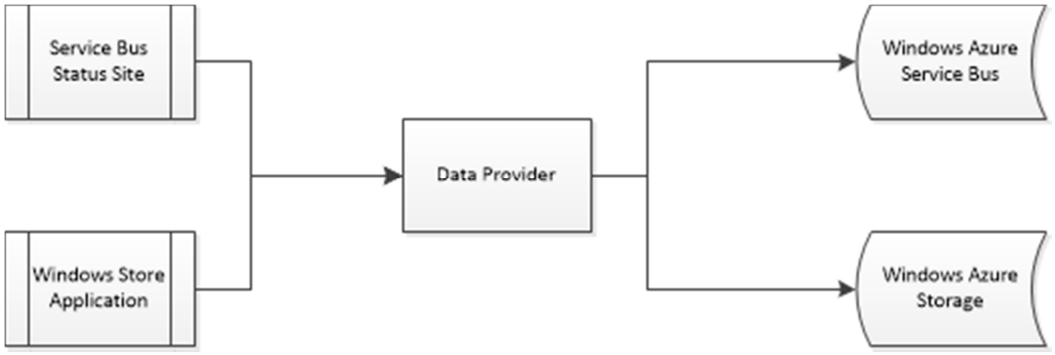

Figure 1: Interaction between applications.

6.1.1 Data Provider

The purpose of the Data Provider is to provide current data for the Windows Store application and both current and historical data for the website (the Service Bus Status Site). Current information can be retrieved directly from the Service Bus and historical information is logged by EF1INT to a table in a Windows Azure Storage account. The current information can be retrieved quickly, but the historical information takes a long time to retrieve. If the historical information was retrieved on command, the users would have to wait a long time.

To make it faster for the users, the Data Provider is made as a Windows Azure Worker Role (a simple application with an endless loop) that retrieves historical information from the Storage account once a day, and keeps it in memory. Because access to the Data provider from the Windows Store Application and the website will only happen occasionally, the Worker Role hosts a WCF service that the information can be retrieved from. Current information, from the Service Bus, is retrieved by the Data Provider when the WCF service is consumed.

6.1.2 Service Bus Status Site

The purpose of this site is to show information about the performance of customers' message queues in the service bus, both current and historical. A service bus administrator can see information about all message queues and customers can see their own message queue. It shows how many messages were in their queue at what time, and for administrators, it also shows the size of the queue as a percentage of how big it can be. An example of what a customer can see is shown in *Figure 2* below.

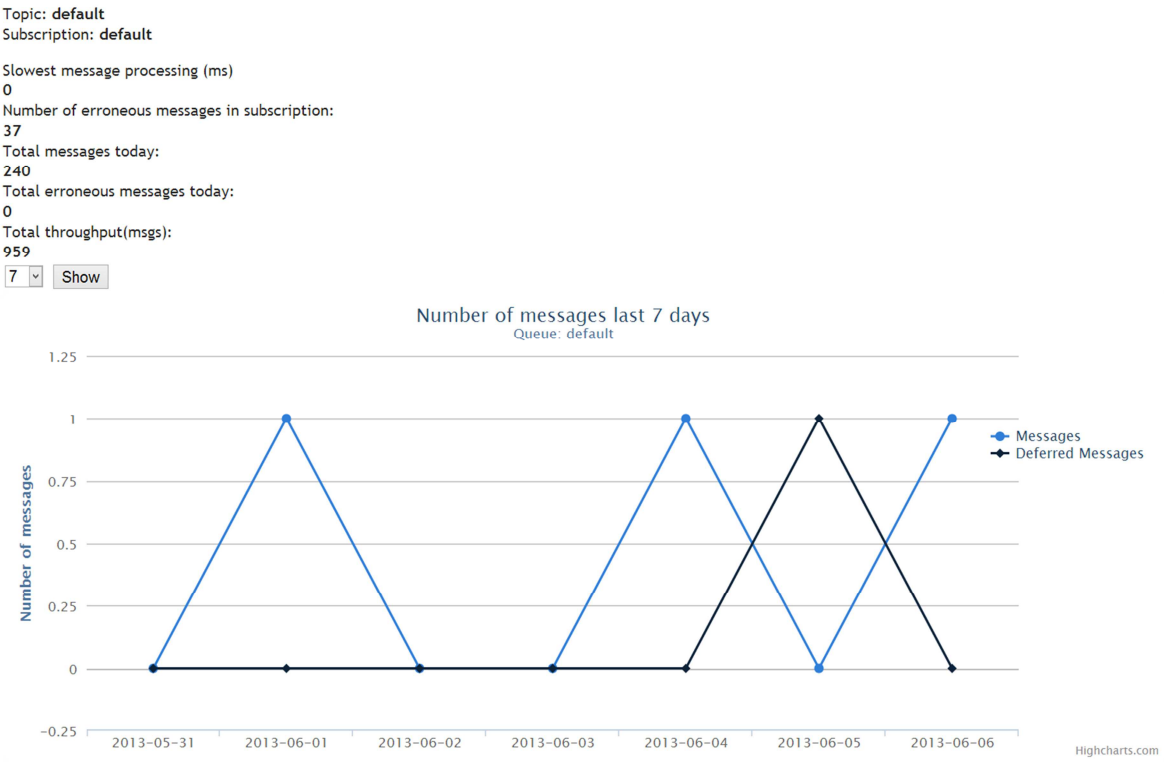

Figure 2: Example data shown in the Service Bus Status Site

The site is implemented as an ASP.NET web page and a WCF client. For graphic visualization of data, a library called Highcharts is used. Highcharts provides a wide range of HTML5/Javascript-based interactive charts. All information shown on the site is provided by our Data Provider. To give the users a better experience, the different elements on the site are updated individually using AJAX as soon as their data is received.

6.1.3 Windows Store Application

The purpose of this application is to provide an overview of the current status of a customer's service bus queue. It is a tile in the Modern UI that is used in Windows 8 based products.

The tile shows information about the customer's service bus subscription (shown in *Figure 3*). The information found on the tile is:

- Name of the subscription
- How many messages it contains
- How many messages produced an error while processing

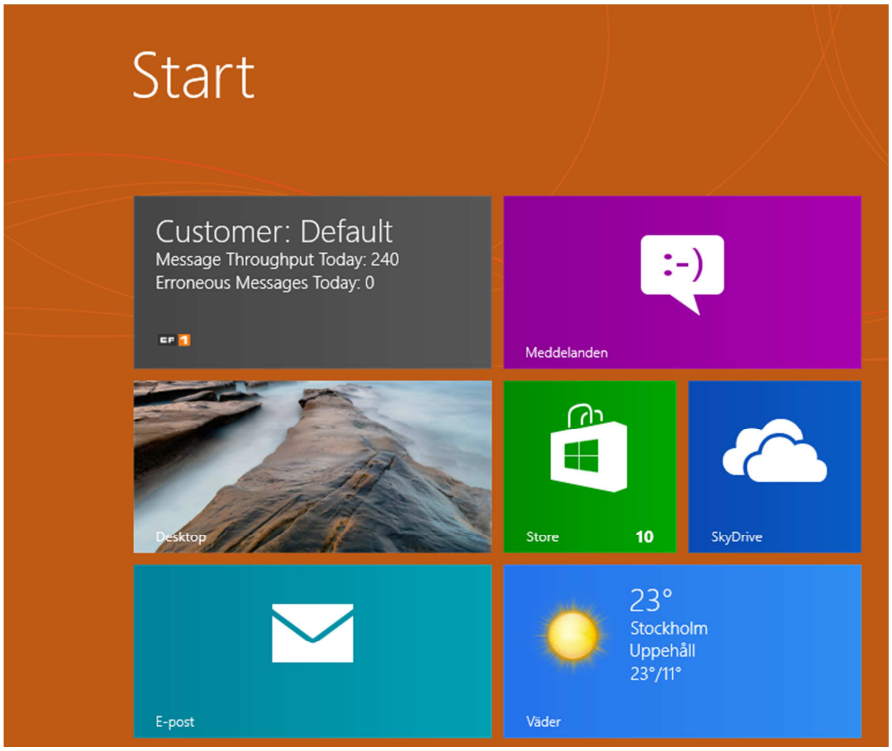

Figure 3: Information about the customer Default is visible on the Windows Store Application Tile.

The tile has a background task that connects to the data provider and updates the information every 15 minutes. To keep the information customer specific, credentials for each company has to be provided in a configuration file. The tile also serves as a link to the Service Bus Status Site.

6.2 Monitoring Helper

The purpose of the monitoring helper is to complement New Relic with support for monitoring Azure storage accounts and Azure service bus accounts, since this is not supported out-of-the-box. This allows the New Relic service to send alerts when an unwanted change within one of these has happened.

New Relic only sends out notifications to users when error rates are too high or response times are too slow. The notification message tells the user the reason (error rates too high or response times too slow) and what application or key transaction it originates from. Because of this, the best way for the monitoring helper to tell New Relic to send a notification is to use the New Relic Agent API to report custom exceptions.

If a change in the service bus or storage account that requires a user's attention is noticed, a notification needs to be sent to the correct user. Users can only subscribe to notifications from applications and key transactions. Because of this, the only way the helper can tell New Relic to send an alert notification to the correct user, the error needs to be sent within a key transaction (within a specific method in the C# code). For New Relic to see the method as a transaction (and to be tracked as a key transaction), a connection to an external source is needed.

To achieve this functionality, the helper is implemented as a Windows Azure Worker Role (a simple application with an endless loop). The worker role runs both a WCF service and WCF client connected to the service. This way, when a method on the service is invoked, New Relic recognizes this as a transaction, which can then be tracked as a key transaction. The client is a loop that monitors an Azure Storage account and an Azure Service Bus account. Depending on what kind of change in a storage account or service bus account is noticed, different methods on the service can be invoked, and the correct users can be notified. The helper is monitored by New Relic like any other Windows Azure application, which lets it report changes using the New Relic Agent API.

7. Conclusion

One of the goals was regarding visualization of status of the message channel. To achieve this, two applications were developed; a Windows Store Application that visualizes live information about a given message channel on its tile and a website that shows current and historical information. The website allows both customers and Softronic to see current health of individual message channels and also allows Softronic to view the health of the message channel service.

To achieve the goal of implementing monitoring software, research and evaluation of existing monitoring systems had to be done.

By researching monitoring systems, we came to the understanding that there are not many systems with the ability to monitor applications in the cloud. It's also hard to find a monitoring solution with support for all the features you may need.

Using the criteria given from Softronic, three candidates were chosen to be tested and evaluated. None of the system researched had full support for the criteria so the three with the most support and with best possibility to achieve the goal was chosen. While evaluating the three, New Relic seemed to be the best one, and it had an API, allowing it to support 100% of the criteria.

New Relic allows Windows Azure application and host monitoring. The basic support from New Relic contains uptime checks, host workloads, application flow-time and user notification by email and/or SMS. To allow for more Azure specific monitoring, like Service Bus and Table Storage monitoring, an application that adds this support to New Relic, the Monitoring Helper, was developed using the New Relic API.

8. Recommendations

When using existing software like New Relic, one must have in mind that there are limitations for these kinds of services; they may not include features that are essential to your application. If there is no current monitoring system with support for the wanted feature, you may want to implement your own monitoring on just that feature or extend a current monitoring system with the feature. For example, if using New Relic, custom metrics can be sent to the service to show the information needed, or the service could be told to send a notification for a specific change in the application that is not necessarily an exception.

Should one consider using New Relic as monitoring service, trying to implement it after the development of the application/service is finished could be inconvenient. Instead one should implement the monitoring while developing the product or at least have in mind that monitoring may be needed at a later stage, and adapt the development accordingly.

When choosing how to store information, one should have in mind that different storage options are good for different things. Windows Azure Table Storage is very efficient for storing large amounts of data, but very inefficient when the data has to be retrieved unless the key for the specific data is known. Because of this, if the data stored has to be accessed on a regular basis, one should consider using a storage option that is more efficient for retrieval of data.

References

[1] Softronic, Våra e-tjänster
http://www.ef1.se/Svenska/Vara-e-tjanster/Sidor/Start.aspx
[Accessed 15-05-2013]

[2] Eric Knorr, Galen Gruman, What cloud computing really means
http://www.infoworld.com/d/cloud-computing/what-cloud-computing-really-means-031?page=0,0
[Accessed 14-05-2013]

[3] Jonathan Strickland, How Cloud Computing Works
http://computer.howstuffworks.com/cloud-computing/cloud-computing.htm
[Accessed 14-05-2013]

[4] Priya Viswanathan, Cloud Computing - Is it Really All That Beneficial?
http://mobiledevices.about.com/od/additionalresources/a/Cloud-Computing-Is-It-Really-All-That-Beneficial.htm
[Accessed 15-05-2013]

[5] Microsoft, What is a Storage Account?
http://www.windowsazure.com/en-us/manage/services/storage/what-is-a-storage-account/
[Accessed 14-05-2013]

[6] Microsoft
http://www.windowsazure.com/en-us/develop/net/how-to-guides/table-services/
[Accessed 14-05-2013]

[7] Alan Smith, Service Bus Messaging
http://www.cloudcasts.net/devguide/Default.aspx?id=11040
[Accessed 14-05-2013]

[8] Microsoft, What is a Storage Account?
http://www.windowsazure.com/en-us/manage/services/storage/what-is-a-storage-account/
[Accessed 14-05-2013]

[9] Larry Franks, Rama Ramani, Monitoring a Windows Azure Application
http://msdn.microsoft.com/en-us/library/windowsazure/hh694039.aspx

[10] Microsoft, What Is Windows Communication Foundation
http://msdn.microsoft.com/en-us/library/ms731082.aspx
[Accessed 14-05-2013]

[11] Microsoft, Client Monitoring Using Agentless Exception Monitoring in Operations Manager
http://technet.microsoft.com/en-us/library/hh230748.aspx
[Accessed 14-05-2013]

[12] New Relic, New Relic Feature Matrix
http://try.newrelic.com/rs/newrelic/images/NewRelic-FeatureMatrix-2011-12.pdf
[Accessed 16-05-2013]

[13] Dell, Foglight
http://www.quest.com/foglight/
[Accessed 16-05-2013]

[14] Microsoft, System Center 2012
http://technet.microsoft.com/en-us/library/hh546785.aspx
[Accessed 14-05-2013]

[15] Microsoft, SCOM Features
http://www.microsoft.com/en-in/server-cloud/system-center/operations-manager-features.aspx
[Accessed 14-05-2013]

[16] Microsoft, System Center 2012
http://www.microsoft.com/en-us/server-cloud/system-center/default.aspx
[Accessed 14-05-2013]

[17] Microsoft, System Requirements for System Center 2012 SP1
http://technet.microsoft.com/en-us/library/jj628205.aspx
[Accessed 14-05-2013]

[18] Microsoft, System Requirements: System Center 2012 SP1 - Operations Manager
http://technet.microsoft.com/en-us/library/jj656654.aspx
[Accessed 14-05-2013]

[19] New Relic, New Relic .NET Agent
https://newrelic.com/docs/dotnet/AgentDocumentation
[Accessed 14-05-2013]

[20] New Relic, Custom Metric Collection
https://newrelic.com/docs/instrumentation/custom-metric-collection
[Accessed 14-05-2013]

[21] New Relic, Transaction Traces
https://newrelic.com/docs/features/transaction-traces
[Accessed 14-05-2013]

[22] New Relic, Cross Application Traces
https://newrelic.com/docs/features/cross-application-traces
[Accessed 14-05-2013]

[23] New Relic, New Relic .NET Agent Custom Metrics
https://newrelic.com/docs/dotnet/CustomInstrumentation
[Accessed 14-05-2013]

[24] New Relic, Key Transactions
https://newrelic.com/docs/site/key-transactions
[Accessed 14-05-2013]

[25] New Relic, How does real user monitoring work?
https://newrelic.com/docs/features/how-does-real-user-monitoring-work
[Accessed 14-05-2013]

Dictionary

IIS - Internet Information Services, Microsoft way to provide internet content such as web sites and WCF-Services.

.NET - Framework developed by Microsoft.

REST - A web API design model.

SHS - Standardized protocol for information exchange where great security and reliability is needed for example authorities.