Interactive graphics on large datasets drives remote condition monitoring on a cloud

Simon Hickinbotham1,2, James Austin1,2 and John McAvoy2

1 Computer Science Department, University of York, Deramore Lane, Heslington, York YO10 5GH, UK
2 Cybula Ltd, Computer Science Building, University of York, Deramore Lane, Heslington, York YO10 5GH, UK
sjh@cs.york.ac.uk; simon@cybula.co.uk

Abstract. We demonstrate a new system for condition monitoring using the cloud. The system combines state of the art pattern search capability with youShare, a platform that allows people to run compute-intensive research in an ordered manner over the internet. Data from sensors distributed across one or more assets at one or more sites are uploaded to the cloud compute resource. The uploading triggers the deployment of a range of pattern search services, and is capable of rapidly detecting novel patterns in the data. The outputs of these processes are archived as a matter of course, but are also sent to a further service which processes the data for remote visualisation on a web browser. The system is built in Java, using GWT and RaphaelGWT for graphics rendering. The design of these systems must satisfy conflicting requirements of data currency and data throughput. We present an evaluation of our system that involves processing data at a range of frequencies and bandwidths that are commensurate with commercial requirements. We show that our system has the potential to satisfy a range of processing requirements with minimal latency, and that the user experience is easily sufficient for rapid interpretation of complex condition monitoring data.

1. Introduction
This paper describes work within the Condition Monitoring on A Cloud project (CMAC) that brings together the process of condition monitoring with the new platform of cloud computing to deliver an integrated monitoring service that combines monitoring information existing in previously disparate solutions, deploys it across geographical locations and so delivers a more powerful solution at lower cost.

Cloud computing [1] is the provision of computing on demand from a remote computer. One aspect of the cloud is the provision of software as a service over the internet, i.e. providing applications (services) hosted remotely. Ideally, these services do not require end-user knowledge of the physical compute resource they are accessing, nor particular expertise in the use of the service they are accessing. Whilst the cloud offers opportunities for remote monitoring of assets, there are issues regarding data volume and data rendering that must be addressed to make the approach effective.

In this contribution, we present an overview of the design of CMAC. The system operates in a manner that combines state of the art pattern search capability with a software-as-a-service platform, called youShare (www.youshare.ac.uk) [2]. In a typical application (described below), asset data may be distributed across a number of sites. CMAC provides a platform that allows
people to run compute-intensive research in an ordered manner over the internet. Data from sensors distributed across one or more assets at one or more sites are uploaded to the cloud compute resource, where condition monitoring (CM) takes place.

The compute resource maintains a scheduler service which recognises data uploads and deploys pattern search services on the data. The pattern search services form the core of the condition monitoring process, and are currently based around AURA-Alert [3] software, capable of rapidly detecting novel patterns in the data. However, since CMAC is built upon youShare there exists the potential to use any software to process the data, as long as the operating system that hosts the software can be run as a virtual machine. The outputs of these processes are archived as a matter of course, but are also sent to a further service which processes the data for remote visualisation on a web browser. The core of the system is built in Java [4], using GWT [5] and RaphaelGWT[6] for graphics rendering, but software written in other languages can be “wrapped” into the system and deployed using virtual machines (VMs). This is a move away from the “one size fits all” approach to mainstream grid or cloud computing, and allows lightweight construction of data flows without excessive need for reconfiguring and rewriting software to make the system workable.

For remote condition monitoring using large datasets, it is often preferable and sometimes essential to present the user with interactive graphics to allow the data to be explored effectively and so to trace any fault. Recently, the JavaScript graphics DOM has made it possible to develop interactive graphics that work on the majority of web browsers on a cross-platform basis. Within CMAC, a combination of pre-rendered graphical objects and asynchronous requests for raw data allow a richly interactive rendering of the data to be made within acceptable processing times.

The design of these systems must satisfy conflicting requirements of data currency and data throughput. In the following sections, we present an overview of our system that involves processing data at a range of frequencies and bandwidths that are commensurate with commercial requirements. We show that our system can satisfy a range of processing requirements with minimal latency, and that the user experience is easily sufficient for rapid interpretation of complex condition monitoring data.

2. Local Condition Monitoring

In this section, we develop a simple model of a generic, local CM system, which allows us to demonstrate the benefits that CMAC can offer. An illustration of our top-level concept of monitoring is shown in figure 1. The system is built on an automated pattern search capability, which is is used to detect events in the data. It is important that the automated pattern search can run without supervision after it is configured. The outputs of the pattern search are accessed by a specialist engineer who is responsible for maintaining the accuracy of the monitoring, and a field engineer who is responsible for using the monitoring data to maintain the asset. The specialist engineer is responsible for configuring the pattern search. We use the term model to encapsulate the configuration. Models can incorporate any pattern to be searched for, and include any parameters needed to operate the pattern search. The pattern search can then run, and it produces monitor output. It is not uncommon for a specialist engineer to tune the models to deliver the most appropriate monitoring data over several iterations.

The field engineer does not want or need access to the models used to interrogate the data. As long as he can access the monitor data in an appropriate form, he can maintain the asset in an informed manner. CMAC is primarily designed with the field engineer in mind, since the feedback loop from monitoring data back to the modelling process is likely to involve a level of interactivity that is not currently possible due to the need to manipulate large volumes of data. The challenge in CMAC is to develop a capability for viewing large volumes of data on a web browser, and to allow informed decisions about asset maintenance to be made regardless of the location of the field engineer.
Figure 1. The pattern search process at the heart of remote condition monitoring. The pattern search requires data and a search model as inputs, and gives monitoring data as outputs. A specialist engineer is responsible for analysing the monitoring data and configuring the model. A field engineer uses the monitoring data for managing, maintaining and planning the use of the asset. Solid lines indicate automated processes. Dotted lines indicate manual processes.

3. Processing requirements

The processing requirements of CMAC are best understood by exploring a set of use cases in which CMAC would operate. In this section, we explore three different use cases, and discuss the way that CMAC would be deployed for each.

Use Case 1: Large Enterprise

A typical example of a large enterprise deployment of CMAC is as follows. A large electricity provider has a condition monitoring department in one of its plants. This department is responsible for the analysis of data from all the assets the group has. Accordingly, the five specialist engineers in the group have developed a set of pattern search tools for this work. It has become working practise for all the asset managers to send data to the CM group for analysis. This is wasteful, since the data has to be moved around the organisation, and then the expert engineer has to find time run the tool set on it. Importantly, the disparate clients of the specialist engineers all have similar needs. A CM service based on their data needs to be made available, but the clients have no particular expertise in modelling the data.

Given these observations, it is clear that it would be better to have an automated means of running tools on datasets, and have the results presented in an intuitive way. This would save the time and effort required in generating bespoke but regular monitoring reports, and thus free resources for more specialised analyses of specific events. CMAC provides this functionality.

Use Case 2: CM service provider

For smaller enterprises, it is not unusual for condition monitoring to be contracted out to domain experts. In this case, the development of tool sets for monitoring can be carried out by the contractor. The CMAC model still works here, but the difference with Use Case 1 is that the private cloud is maintained by the contractor. We still see the separation of the model-building, which becomes the contractor’s responsibility, from the condition monitoring, which forms the basis of the service that the contractor offers. Note that there are issues with the privacy of sensitive data (sometimes share-price sensitive data) that must be overcome in this use case.
Use Case 3: Add-on to data warehousing service

The final use case considers CMAC as an add-on to the offering of a data warehousing provider. The service scenario is similar to that for use case 2, but with the emphasis on service shifted to data curation rather than analysis. It can be assumed that data security is not a new issue in this use case, and that CMAC would provide a new revenue stream to the service provider, rather than a more efficient implementation of an existing service as in use case 2.

4. Overview of the CMAC design

In order to develop CMAC, we required a design that accommodates the following workflow:

(i) **Data** will be continually uploaded from the assets being monitored to a cloud.
(ii) Specialist Engineers will develop **Models** of the signals that need to be detected, and upload these to the cloud.
(iii) **Services** on the cloud (making use of software libraries) will apply the models to the data, to create output data.
(iv) **Views** on the outputs can be served to browsers, to allow mobile monitoring by field engineers.

Figure 2 gives a top-level overview of the relationship between entities in the CMAC process. There are five boxes in the diagram. The central box represents the central compute resource, which will be a cloud computing platform running YouShare. The entire process is managed from this system, using a variety of processes that will be discussed below. There are four elements interacting with the distributed compute service. To the top of the diagram, we have the software libraries that form the basis of the monitoring service, including the scheduler, pattern recognition and view generation services. To the bottom, we have a box representing the data. Note that there are three types of data (not including associated metadata): raw data is the monitoring information that is collected from the asset; model data is the specification of the means to search the raw data; monitoring (output) data is the result of the application of the model data to the raw data.

![Figure 2. Overview of CMAC concepts. A distributed compute resource manages, data, search models and software libraries to deliver a remote monitoring service.](image-url)
To the left of the diagram is a box labelled “Search Model Generation”. It is anticipated that significant expertise is needed to generate the models that are used to monitor the data. Recall that within CMAC, model generation is distinct from condition monitoring. Models are used to generate “views” of the asset using the data from the sensors. Finally, the box to the right encapsulates the process of accessing the cloud to receive reporting of the condition of assets from remote locations. This will form the CMAC service for most users.

With the top-level design to hand, we detail below the existing technologies that are used to build CMAC.

4.1. YouShare

YouShare is the platform for distributing software applications as services across computing resource. It has been developed at the University of York for sharing data and software in the academic community (www.youshare.ac.uk). A summary diagram of the YouShare system is given in figure 3. The basic YouShare model is as follows. All interaction is via a web-based Portal. Within YouShare, requests from the Portal are handled by Servlets, which create YouShare job requests that are passed to the a Broker. The Broker handles the negotiations between a storage subsystem (or SRB, the storage resource broker) and the Compute Farm, that marries data and software with appropriate hardware resource to process the job as efficiently as possible. The compute farm is capable of running any operating system within a virtual machine, and thus provides the capacity to easily run uploaded software without needing to port it to another OS, thus reducing the programming overheads that would normally be required when sharing software. When the job has completed, the Broker also carries out any clean-up tasks and returns the result to the servlets that repackage the data in a form compatible with the Portal.

4.2. Signal Data Explorer (SDE)

SDE [7] is Cybula’s time-series pattern matching engine and data visualisation tool. It works as a stand-alone product on the Windows operating system. It has functionality for: visualising and exploring time-series data sets; creating and manipulating patterns as search queries (models); using models to search for matches in the data; using tools to manipulate data based on the results of a search. CMAC will run components of SDE as a service, making this functionality available for remote monitoring on any HTML5 browser from any operating system.

4.3. AURA-Alert

AURA-Alert is a novelty-detection plug-in for SDE. It is designed to detect anomalies in signal data from complex assets. It may be used on historical and real-time data feeds. It is particularly simple to use and set up and scales to large data volumes. Building a remote monitoring

---

Figure 3. YouShare overview.
service involves deploying AURA-Alert on the cloud within CMAC, and will explore the different modelling and data-representation issues between AURA-Alert and SDE.

4.4. CMAC data flow

Our implementation of CMAC is presented in figure 4. This combines the concept from figure 2 with the YouShare model from figure 3. Starting from the left of the diagram, we have indicated the various classes of interaction with the CMAC service. From top to bottom, the three classes are Monitor, Model, and Data. When carrying out condition monitoring, the user accesses the cloud to retrieve information regarding the condition of assets. The other two interaction types only interact with CMAC in order to upload Models and Data. We envisage that models will be created and uploaded manually, whereas it is possible that the uploading of data could be completely automated.

When new data appears, the CMAC cloud then carries out condition monitoring using the models and the data. Each arrow inside the CMAC cloud represents a process. Each process will require brokering, servlets and metadata. These have been removed from figure 4 for clarity.

Within The CMAC cloud, the SRB houses the various repositories for data, models, outputs and views. It is likely that the compute services will also be archived within SRB. We have also omitted this for clarity. The processing of data is carried out on various machines within the cloud, represented here by the “Compute” box. The hardware underlying this processing can take many forms, and be located at one or more sites. This will be invisible to the user once the protocol for deployment of services has been set up. We have divided the CMAC process into two main tasks: Pattern Search, and View Creation. This strategy allows us best to deal with the very large data overheads that may be encountered in asset monitoring. The idea is that pattern search is the most intensive process, which must be continuously applied to the data streams that come from the assets. In addition, some pre-rendering of data for rapid interactive viewing will also be necessary, and is carried out immediately after the data is processed. Note that CMAC can be configured to have sufficient virtual machines available perpetually, with services ready to handle data as it is uploaded. This means that there is very little overhead in using CMAC for continuous monitoring compared with a standard web service model.

The solution outlined here is appropriate for all the use cases discussed in section 3. In a large enterprise (see use case 1), the whole system could be run in-house provided the pattern search software is licensed and managed appropriately. Use cases 2 and 3 would require additional interfacing to maintain information security. This issue is common to Data Historian service providers, so we anticipate that finding an appropriately secure solution would not be overly problematic.
5. Compute tasks
The application of models to asset data must be carried out as sufficient data enters the CMAC cloud on scheduled intervals. Table 1 shows an example of a scheduling table that manages this process. In the example there are six assets over two sites. Three models are deployed for condition monitoring across these assets. Model two is deployed across both sites. The scheduling table would be used within the cloud to broker the processing jobs across the cloud. This approach highlights the main risk in CMAC, which is the handling of the networking overheads. In some cases, it would be appropriate to co-locate processing hardware with major data sources to minimise networking requirements. The data broker should be configured to distribute jobs around the cloud appropriately.

6. Web-based visualisation of complex data
CMAC is not practical without interactive graphics, since field engineers need to explore monitor data to gain full understanding of the condition of an asset. Once data is processed, the raw outputs of the pattern search are placed in a dedicated database in the SRB based storage system. Views of these outputs can then be created in a separate process, and served to the monitoring portal, using the “Create View” service as shown in figure 4. Since it is likely that many end users will require access to the same view, the views can be stored in a database on SRB, and accessed via the portal. Note that the creation of new views on-demand may be possible in some cases.

6.1. Existing technologies for developing interactive graphics
Only a few years ago, interactive graphics were very difficult to achieve across all browsers due to the differences in standards available at the time. With the emergence of HTML5, it is now

![Figure 4. CMAC overview. Information flows roughly anticlockwise from models and data back through to monitoring. Note separate storage of output and view data, and direct access of view data from the monitor portal.](image-url)
much more straightforward to develop interactive graphics without the need to re-code the same applications for the popular browsers. The solution we are working on in for CMAC is based on the following four software technologies.

(i) **AJAX:** (Asynchronous JavaScript and XML) is a technique that allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. By using AJAX, it is possible to update parts of a web page without having to reload the whole page. AJAX will be useful to CMAC, since it provides a mechanism to fetch updates to individual views from the server without having to reload whole web pages.

(ii) **Google Web Toolkit (GWT)** [5] is an open source set of tools that translate native Java code into client-side JavaScript browser applications. It is licensed under the Apache License version 2.0. GWT emphasizes reusable, efficient solutions to recurring Ajax challenges, namely asynchronous remote procedure calls, history management, bookmarking, internationalization and cross-browser portability.

(iii) **Raphael:** (from wikipedia) is a JavaScript library that draws vector graphics for web sites. It is able to deploy SVG for most browsers, but can also use VML for older versions of Internet Explorer. Raphael currently supports Firefox 3.0+, Safari 3.0+, Opera 9.5+ and Internet Explorer 6.0+. Raphael is a low-level graphics library. The advantage of a low-level approach is that we can more closely align the functionality of the browser-based interface with the monitor data.

(iv) **RaphaelGWT** [6] is a wrapper around the Raphael JavaScript library that allows Raphael objects to be manipulated via GWT. This allows us to place the graphics we require inside the YouShare framework, since youShare was developed using GWT.

![Figure 5](attachment:image.png)

**Figure 5.** Demonstration of asynchronous rendering of multivariate data in a web browser. The CMAC system first fetches summary plots for each of the variables being monitored, then the user selects variables of interest by clicking on them. This triggers a remote procedure call which fetches and plots data at high resolution. The view of the data 0.5 seconds after sending the request is shown on the left. The view of the data after three plots have been selected for further examination is shown on the right.
By combining these technologies within the software and hardware infrastructure described above, it is possible to render views on CM data remotely in a usable fashion.

6.2. Developing usable interactive graphics

In this section, we demonstrate how a combination of asynchronous rendering and a dual representation of monitoring data can form the basis of interactive exploration of monitor data on a web browser. The challenge is to make browser-based remote monitoring usable. Usability research [8] has found that users find that waiting more than 2 seconds for a web page to load reduces the effectiveness of the medium. Yet CMAC requires that large volumes of monitoring data be made available at the resolution of the individual datum.

We argue that a dual representation of data is the best way to meet the conflicting requirements of fast rendering and high fidelity resolution. By taking this approach, initial “snapshot” views of asset condition over a time period can be rendered quickly. The field engineer can then be familiarising himself with this global view whilst more detailed information is retrieved from the server as a background process.

To test this approach, we parsed the output of an AURA-Alert service into two data files. The first of these files consisted of summaries of the fluctuations in the variables over time as SVG paths, scaled to fit the viewing window. The second file is a binary file of the raw monitor data at full resolution. The file is appropriately indexed to allow fast seeking of data. These files were stored on an off-site server, accessed via the internet. A GWT/Raphael program managed the remote procedure calls to fetch this data from the server and render it on a browser.

There were 17 variables used as monitoring data in the study, each of which contained 30,000 individual data points. Plots of each variable were fetched asynchronously, so that they could be rendered as soon as possible by the browser. The left-hand panel of figure 5 shows the plots 0.5 seconds after the web page began loading. Seven of the seventeen plots have already been rendered. The remaining ten plot were rendered within a further 0.5 seconds. (We have found that the first few plots take more time than subsequent plots to fetch from the server, probably due to new channels being opened having a significant time overhead).

The advantage of rendering in SVG is that the graphics are interactive. This means that the user can select individual plots in the top-level view and examine them at higher resolution. Here, the second file format is used, which contains the entire dataset in binary form. The vertical green bar in the left panel of figure 5 can be dragged to a region of interest. When the user releases the bar, the browser initiates a remote procedure call to fetch the high-resolution data from the binary file for whichever plots have been highlighted. These can be fetched in under 0.5 seconds. By moving the mouse over these plots, the values of individual data points can be displayed for interpretation. Note that we tested this on Firefox, Internet Explorer and Safari, on Linux, Windows and Apple operating systems. All performed equally efficiently.

7. Conclusion

Cloud computing offers an opportunity to make the results of condition monitoring readily available to a range of stakeholders responsible for the maintenance of an asset. In this document, we have detailed the main challenges to be faced when implementing such a service. The main innovation has been to find a reliable means of rendering large volumes of data across the key modern browsers with sufficient resolution and speed.

Having developed the core technology, we intend to investigate a range of usability issues in remote browsing. Some principles are well known, for example Edward Tufte’s sparklines [9], but the way these principles are used in new media is an active area of research. CMAC provides a rich and varied source of CM data to develop these ideas in an interactive setting.

We are approaching a future in which process control systems will involve the linking of personnel, data and site sensors together in a more distributed manner [10]. Whilst the use
of the internet for exchanging expertise and knowledge is widely known, the development of systems such as CMAC offers the provision of applied site monitoring as the basis for the rapid connection of real data with distributed expertise on a large scale.

8. Acknowledgements
This work was funded by the Technology Strategy Board, project ref: 5175-33386, Cybula Limited and the University of York.

References
[1] Armbrust M, Fox A, Griffith R, Joseph A D, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I and Zaharia M 2010 Commun. ACM 53(4) 50–58
[2] Austin J, Fletcher M, Jackson T, Jessop M, Turner A and Weeks M 2011 UK e-Science All Hands Meeting (AHM)
[3] Austin J, Brewer G, Jackson T and Hodge V 2010 Proceedings of 7th International Conference on Condition Monitoring and Machinery Failure Prevention Technologies: (CM 2010 and MFPT 2010) vol 1 pp 699–711 ISBN 978-1-61839-013-4
[4] http://www.java.com/en/
[5] http://code.google.com/webtoolkit/
[6] http://code.google.com/p/raphaelgwt/
[7] Fletcher M, Jackson T, Jessop M, Liang B and Austin J 2006 Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid CCGRID ’06 (Washington, DC, USA: IEEE Computer Society) pp 217–224 ISBN 0-7695-2585-7
[8] Nah F F 2004 Behaviour & Information Technology 23 153–163
[9] Tufte E R 2006 Beautiful evidence first edition ed (Cheshire, Connecticut: Graphics Press) ISBN 0961392177
[10] Oksanen J 2012 Process Engineering 93 17–20