A Machine Learning Middleware For On Demand Grid Services Engineering and Support

Wail M. Omar [1,2], A. Taleb-Bendiab [1], Yasir Karam[1]

[1] School of Computing and Mathematical Sciences
Liverpool John Moores University
Byrom Street
Liverpool, L3 3AF, UK

[2] Faculty of Applied Sciences
Sohar University
Sohar, P.C. 311
Sultanate of Oman

Abstract. Over the coming years, many are anticipating grid computing infrastructure, utilities and services to become an integral part of future socio-economical fabric. Though, the realisation of such a vision will be very much affected by a host of factors including; cost of access, reliability, dependability and security of grid services. In earnest, autonomic computing model of systems' self-adaptation, self-management and self-protection has attracted much interest to improving grid computing technology dependability, security whilst reducing cost of operation. A prevailing design model of autonomic computing systems is one of a goal-oriented and model-based architecture, where rules elicited from domain expert knowledge, domain analysis or data mining are embedded in software management systems to provide autonomic systems functions including; self-tuning and/or self-healing. In this paper, however, we argue for the need for unsupervised machine learning utility and associated middleware to capture knowledge sources to improve deliberative reasoning of autonomic middleware and/or grid infrastructure operation. In particular, the paper presents a machine learning middleware service using the well-known Self-Organising Maps (SOM), which is illustrated through a case-study scenario -- intelligent connected home. The SOM service is used to classify types of users and their respective networked appliances usage model (patterns). The models are accessed by our experimental self-managing infrastructure to provide Just-in-Time deployment and activation of required services in line with learnt usage models and baseline architecture of specified services assemblies. The paper concludes with an evaluation and general concluding remarks.

1 Introduction

Due to the pervasive distributed computing environment, such as information systems and computational Grid, has enabled a new generation of applications that are based

M. Omar W., Taleb-Bendiab A. and Karam Y. (2005).
A Machine Learning Middleware For On Demand Grid Services Engineering and Support.
In *Proceedings of the 2nd International Workshop on Computer Supported Activity Coordination*, pages 89-100
DOI: 10.5220/0002558200890100

on seamless access, aggregation and interaction. The dramatic side of the story is a strong presence of the plea that those decentralized Grids are potentially affected by number of primitives derived from their anatomy, is that they are inherently large, complex, heterogeneous and dynamic, globally aggregating large number of independent computing and communication resources. This had clearly exposed an essence exigency for a vital change of how these applications are developed and managed, which has motivated researchers to consider other techniques used by biological systems to deal with complexity, dynamism, heterogeneity and uncertainty, which is referred to the autonomic computing.

In particular, autonomic computing research is exploring and developing models to support distributed systems lifetime management and unpredictability by delegating many of the systems management, maintenance tasks to the software itself including; resource management, job scheduling, services failure prediction, load-balancing, QoS, services reservations, deployment and discovery [1].

A prevailing design model of autonomic computing systems is one of a goal-oriented and model-based architecture, where rules elicited from domain expert knowledge, domain analysis or data mining are embedded in meta-systems (self-management) software systems [2] to provide autonomic systems functions including; self management, self optimization, self-tuning and/or self-healing. Whilst, such a rule-based approach is reported to be appropriate for systems self-management with inherently stable operating rules (and/or policies), it suffers from the lack of support for variability and evolution of domain operating rules and policies. Thus, requiring at best a manual rule-base maintenance or at worst software systems maintenance.

Among tremendous machine learning deployment; in which a model consisting classes and features is to be trained adaptively to preserve data learnt , this model can then be used to perform taxonomy against new epochs, another approach of anticipated properties of system is yet claim more advantages of neural networks in learning machine symptoms over time.

Motivated by research on unsupervised machine learning techniques, which can exploit the abundant systems' monitoring data; this paper advocates for the need for a machine learning utility and associated middleware to capture/extract and evolve knowledge models (sources) from the infrastructure operating systems. Such knowledge/models can then be used by our developed autonomic middleware control services.

In particular, the paper will present a machine learning middleware service using one of the well-known neural networks unsupervised learning techniques Self-Organising Maps (SOM) to nominate classification and clustering applied in an on-demand home-networked appliances scenario. In which, the SOM service is used to classify types of users and their respective networked appliances usage model (classes/features) and services dependencies. The models are accessed by our experimental self-managing infrastructure to on-demand and Just-in-Time deploy and activate required services in line with learnt/extracted usage models and baseline architecture of specified services federations/assemblies1 and discovering and activating additional services on-demand. Other types of supervised learning algorithms like regression, decision trees, Bayesian learning, reinforced learning and

[1] Discovering of the services through anticipating the required services for each group of users and reserved services for them

many others demand more analyzing to achieve prospected goal with minimum error; this would ultimately claims manual procedures and testing until reaching those goals, which in return requires managed centre administration. The paper will finish with some concluding remarks and outlines further works.

The remainder of this paper is divided as follow: Section 2 outlines the motivations for the work including related work. Section 3, describes the intelligent connected home machine scenario based on demand services. Section 4 introduces an application of classification method to support on demand grid service binding, activation and reservation. This is illustrated though a simplified scenario of a SOM-based middleware service for an intelligent connected home machine in section 5. Sections 6, presents an implementation for services and job schedule services using simple example. The last section is the conclusion and future work.

2 Motivations and Related Work

Many applications of the autonomic computing model in grid computing settings have been widely reported by major IT players including; Sun Microsystems, Hewlett-Packard and IBM as a way forward for a highly automated computing systems [3]. By creating hardware and software that can diagnose and solve network problems, thus improving high-availability while reducing IT operation costs [3].

A plethora of publications are now available ranging from a description of the general benefits of autonomic computing, case-studies to design models, to tools and techniques for design, deployment and management of autonomic computing software systems.

Much research is now underway adopting the using of machine learning to support different tasks of autonomic grid computing such as self-management, self-configuration, self-protecting, and other general QoS improvement. For instance, M. Chen *et al.* [4] report on their application of the C4.5 decision tree algorithm and data mining to categorize causes of failure in large Internet sites such as eBay. Many of researchers have recognized the importance of using autonomic system for path failure as one cause of services failure and to improve the QoS. G. Candea, *et al.* [5] presents an Automatic Failure-Path Inference (AFPI) as an application-generic and automatic technique for dynamically discovering the failure dependency graphs of componentized Internet applications. They focused on applying AFPI to applications built on Java 2 Enterprise Edition middleware. AFPI-generated f-maps correctly omit dependencies that appear in the static call graph but do not result in observed fault propagation at runtime. The accuracy of applying autonomic system using machine learning or data mining algorithms for large, distributed, and dynamic application environments is one of the critical problems. M. Chen *et al.* [2] present a dynamic analysis methodology that automates problem determination in these environments by 1) coarse-grained tagging of numerous real client requests as they travel through the system and 2) using data mining techniques to correlate the believed failures and successes of these requests to determine which components are most likely to be at fault. They implemented Pinpoint, a framework for root cause analysis on the J2EE platform that requires no knowledge of the application components. In large scale system, there is an expectation for large number of failure services; this produces the

demand for failure management system. M. Chen *et al.* [6] present a new approach to managing failures and evolution in large, complex distributed systems using runtime paths. They use the paths that requests follow as they move through the system as their core abstraction, and their "macro" approach focuses on component interactions rather than the details of the components themselves[2].

In our approach, we use Self-Organizing Maps (SOM) to underpin autonomic middleware services, in that, in this research, to do the classification process among groups of consumers according to the usage services. If the middleware gets this classification, then it can predict the required services for a group of consumers according to their assemblies. The middleware acts to reserve and prepare the required services for the consumer before prior to the demand request; this will add synchronization of service usage. Of course the reservation services may not require one, but the accuracy and responsiveness of the system will increase with the learnt services usage by consumers, which will in turn increase the training process for the system. Job schedule also gets benefit from classification services, because the middleware can anticipate the peak load and then manage the job schedule according to it[3].

3 On-Time Intelligent Connected Home Scenario

Grid computing requires a range of management processes and services for making the middleware or broker interactive with the services/infrastructures and applications faster. On-time services are another way to reduce the interaction between the consumers and the devices by adjusting them to give better and fast services to the consumers. Such processes and services to work fast and reliable need an autonomic services or management services to run inside the middleware. Such services will be one of the core units for the middleware in doing the automated jobs for the Open Grid Services Architecture (OGSA) [13]. Taking benefit from the concept of service-oriented model of the grid, the consumer should consider his requirements in advance before invoking the demanded services to reduce the unnecessary invocation process, and hence reduce the response time, or in other word is to create an advanced on demand services request. A service reservation is much more agreeable in this case, which is responsible to organize the consumers' requests in advanced. Such service can be integrated as one of the middleware core services. The middleware needs to be more specific in the provision of the services required, clustering and classifying consumers should be yield, and this method of classification is much applicable in our

[2] Efficient performance in grid computing requires mechanisms for managing the load balance, recover services failure and discover most suitable services for the consumer, and improve QoS. Most of these things can be done using instrumentation inside the middleware. This instrumentation needs to be autonomic to tune the reaction of the middleware against different types of changes, and force self-configuring, self-optimizing, self-healing, and self-protecting Middleware needs to get readings from the services or targets.

[3] Our approach for services reservation and job schedule depends on using one of the intelligent classification methods for the system's users.

case while the target of hypothesis is instantly unknown until the final trained model is built.

To elucidate the idea of on-time or on demand services based on user usage classification, connected home devices scenario is adopted. The intelligent connected home machines are the next generation of the home devices, which depend on local and remote services to be available on-time. Each such device may use remote grid services. Each consumer can use a number of services (we will use name services instead of devices). Of-course there are a number of consumers sharing approximately the same devices each time, we tried to classify these consumers according to their usage devices. Reasoner is the intelligent services inside the middleware responsible to do the autonomic stuff, which will be clustering in this case according to the consumer's usages. Figures 1 and 2 demonstrate this idea.

The scenario starts by training the machine learning service with the consumer's devices usage. The sensors inside the connected homes starts collect information for the new consumers and store it in the logger. This information is used by the machine learning service (inside the reasoner) to classify the new users to one of the classified groups. The benefit after classifying the users into group is to anticipate the required services by each consumer beforehand. These anticipating results are fed to the service reservation system with the required information. This system determines the required services, time of operation, required resources, dependency and other things required to run the service for each consumer and send it to the job schedule system. The jobs schedule system at this stage is responsible to do schedule of the services, anticipating the load on each service, and recover the fault tolerance problem before occurrence. Job schedule system is responsible to arrange services readiness before requesting in order to reduce the response time and give better QoS.

Because currently existed grid architecture model lacks classification services, we are proposing a SOM-based classification service to underpin our unsupervised machine learning middleware service. There are many others types of unsupervised learning methods that can be used to do this job, like Support Vector Machine (SVM) [15, 16]. For us, we tried to start with SOM because it is widely used in such problem, and it works efficient with such applications [11, 17]. In the future, SVM will also be test to reach to the best method of consumers' classification.

The data collected from the middleware repository has rich information to be processed, and starting with the definition of self-organizing map method as a vector quantization method which places the prototype vectors on a regular low-dimensioned grid in an ordered fashion [14].

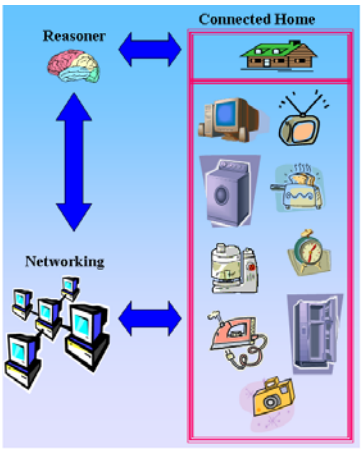

Fig. 1. Illustration of the connected home case study.

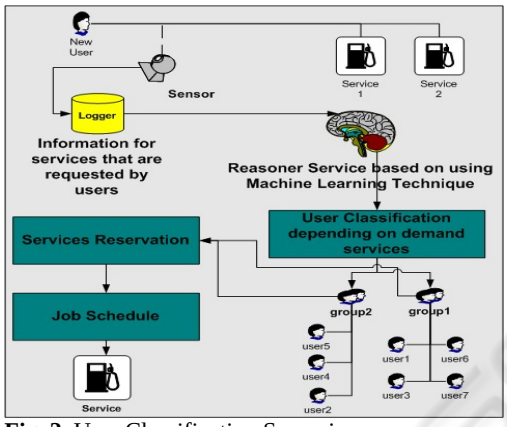

Fig. 2. User Classification Scenario.

Returning to our scenario of using home machine devices, the input data is presented as XML, which can be used in the future with heterogeneous, decentralized and distributed environments. On the other hand, XML can be used to store a large amount of data with small size, and the RAM can easy work with it, therefore we thought in the future XML will be useful to represent the huge amount of data. Figure 3, presents a sample of the XML for the input data that is used to train the system for different cases (i.e. type of devices, category, and time of work).

```
<?xml version="1.0" standalone="yes" ?>
<Categories>
      <Category ID="1">
            <CategoryName>Entertainment</CategoryName>
            <CategoryDescription>Playing</CategoryDescription>
            <Device>
                  <DeviceID>2</DeviceID>
                  <DeviceName>Music center</DeviceName>
                  <DeviceDescription>music 60W</DeviceDescription>
                  <TimeFrom>10</TimeFrom>
                  <TimeTo>12</TimeTo>
                  <DeviceStatus>On</DeviceStatus>
            </Device>
            <Device>
                  <DeviceID>11</DeviceID>
                  <DeviceName>Play Station</DeviceName>
                  <DeviceDescription>Playing TV Game</DeviceDescription>
                  <DeviceStatus>OFF</DeviceStatus>
            </Device>
      </Category>
</Categories>
```

Fig. 3. XML Schema for the input data.

4 SOM Implementation

The concepts of the SOM are out of the scope of this paper, and there are many references on the using of SOM in different applications [9,10,11,12,14].

User's classification using SOM will be a services used by the middleware to do the autonomic work for the grid. The intelligent middleware will be responsible for services reservation, job schedule, and other services which will be described in future papers, like fault tolerance and load balance. The SOM services as web services will be one of the core service of the middleware in doing the autonomic jobs for the Open Grid Services Architecture (OGSA). In our approach we depend on doing consumers' classification using SOM will be part of the middleware which will achieve prediction services over the data repository mined out from consumers usage to the grid in order to tune other middleware activities like service reservation, job schedule, fault tolerance and load balance. Actually the classification service is a part of the autonomy inside the middleware has prioritised role over all other components.

The SOM toolbox for Matlab is an effective software tool for the visualization of high-dimensional data. It converts complex, non-linear statistical relationships between high-dimensional data items into simple geometric relationships on a low-dimensional display [18].

The SOM is initializing using either random or linear initialization. For train the map, SOM uses sequential or batch algorithms by using som_make function, the resulting visual map exhibits the neighbourhood between the neurons and the input training samples updating Best Matching Unit (BMU). The quantization error could be measured using som_quality function which supplies two measures: average quantization error and topographic error. A schematic diagram at Fig 4 [10] illustrates SOM cycle processes, SOM model delivers logic decisions from the visual maps taking benefit from labelling feature in som_autolabel and som_addlabels functions, hence we can build a programming model achieving SOM method and outputting decisions from calculating BMU for a given data vectors using som_bmus function and other related useful functions provided by the toolbox.

VS.Net is used to develop a complete environment for getting training data, scaling data, learning SOM and get the classification results, and simulate the inputs later based on using Matlab function as ASP. This environment will be developing to be one of the middleware services.

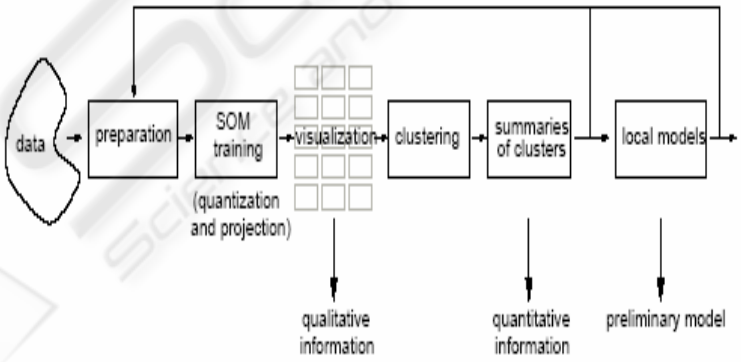

Fig. 4. Using SOM in preparation-survey –cycle [based on 10]

4.1 Data Collection

In order to produce experimental data for the automated classification of users and their home devices' usage models, which will be used by our autonomic middleware for the intelligent connected home a simulation software has been developed using VS.Net and Matlab .

Data format for SOM is a main concern to prepare the samples in proper iterations, and then it will be easy to construct them and build the data structure, which is a Matlab SOM struct using som_data_struct function included with the SOM toolbox for Matlab. Data pre-processing needed can be either simple linear transformations, normalization or logarithmic scaling especially when the divergence of ranges of data is too high, this is done using som_normalize function. After that, the scaled data is used to feed the training system of the SOM.

Different types of categories are selected to represent the different types of home-machine categories, each one of these category contains a number of home-machine devices or services. 0 and 1 are selected to represent the status of the devices as "OFF" or "ON" respectively.

5 Results of SOM Classification for Connected Home Machine

The visual results of the experiments are obtained using an implemented machine learning middleware service. Matlab SOM library [14] is used to implement such experiment. Figures 5, 6,7 and 8 show SOM-based classification results of the our input data generated from Matlab, which represent a simulation of our self-managing middleware for intelligent home networks. The results represent classification for different type of users and devices. Figure 5 shows many correlations between device usages, which are obtained after the training phase which included 1000 input sample data and 10 trainees (devices consumers). Sample of these correlations are described in the following points:

a. Lights and PlayStationII correlates as shown in Figure 5.
b. Video and Coffee Machine correlates as shown in Figure 5.
c. Video CD and Fans correlates as shown in Figure 5.
d. Vacuum cleaner and Washing machine correlates as shown in Figure 5.

Figure 6 represents U-Matrix distribution of labels for the connected home devices. Figure 7, shows shaped U-Matrix with coloured regions exhibiting 7 clear clusters of the map. The critical analysis of this approach depends on selecting and scaling the correct data for training the system.

At runtime, the machine learning middleware service using the training data (user and device classification) can classify log in users according to known users/devices (one of the seven classified region). Each of which is for instance specifies the user types and their uses model such as device usage order and time of usage. One of the applications of such a user device usage model is used for our autonomic services reservation and provision services.

SOM prevail some shortcoming when large quantity of good quality representative training data required training the maps, possibly misleading visualisation when neurons close together in output space represent similar input patterns. The SOM

toolbox comes with some error measures: som_quality which measures quantization and topographic error of SOM, som_distortion which measures SOM distortion and som_distortion for elements of the SOM distortion measure.

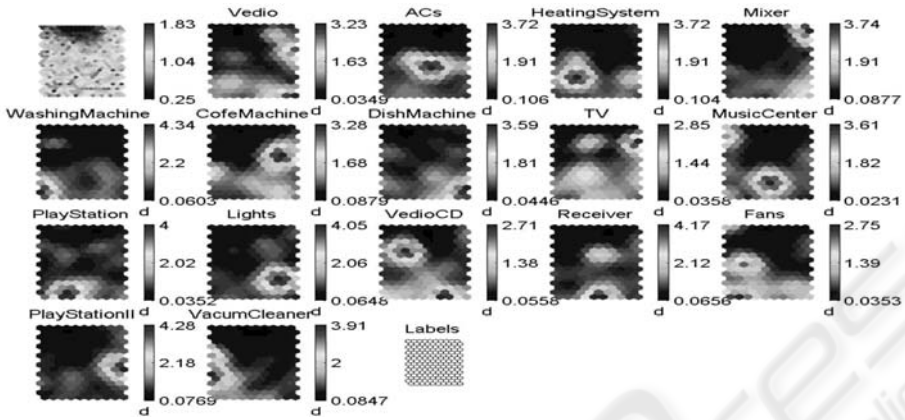

Fig. 5. SOM visual classification

Fig. 6. U-Matrix distribution of labels

Fig. 7. U-map of SOM maps resulted data

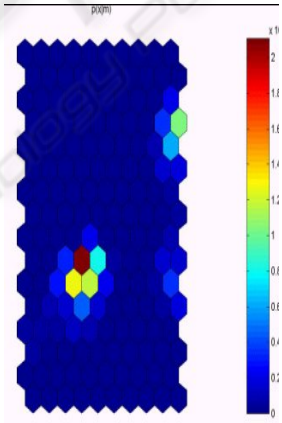

Fig. 8. Probability Distribution Function PDF of the input vectors

6 Implementation of Using SOM Service with On Demand Services Scenario

User classification scenario based on devices usages is used to implement the idea of on demand services as described in section 3. The SOM service is used to find the pattern for each user. SOM is design as web service to be integrated with the core functions of the middleware. The new user is sort to one of the classified groups, then the system send a notification to the service reservation system with this new user and the estimated required service that should be prepare by such system. For example,

figure 9 demonstrates a list of devices required by user ‘Wael’. Also it shows the new users who enter to the system, in this case ‘Wael’ and ‘Taleb’. This information regarding the new users with their requirements is sent to the job schedule services to manage the execution of such services. For example, figure 10 presents the notification of execution service (device) light for user ‘Wael’. It also describes the time of execution for this new users. The system predicts the time for the execution for each service based on gathering information through dynamic instrumentation. Of course, the consumer has the right to change this time or even the service to adjust the system and make it suitable for his requirements.

The software for the services reservation, notification and job schedule for connected home machine has been implemented using VS.Net.

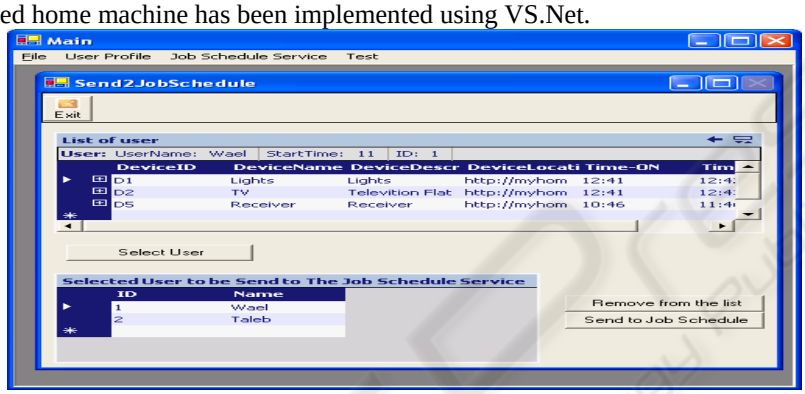

Fig. 9. Service Reservation

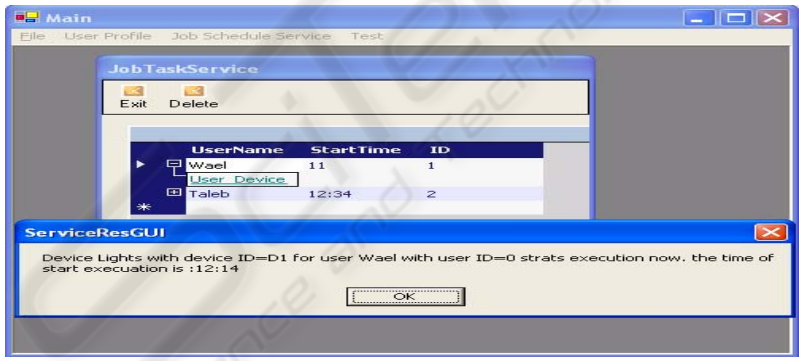

Fig. 10. Job schedule & notification services

7 Conclusions & Future Work

Autonomic computing is addressed to be one of the methods that are used to manage the grid environment. User classification is here proposed to enhance job schedule and services reservation processes based on automated way. Machine learning is used to do the autonomic staff. SOM is suggested to be used to do the classification staff as a service that can be attached later to middleware core services. VS.Net and Matlab functions are used to develop the environment for the user classification process starting by creating the random data as XML or text format, scale the data, train the

map and then predict the type and pattern of the new user. The goal of predicting the new user is to determine the required reservation services. Connected home machine devices scenario is adopted to demonstrate the idea of classifying Users according to the devices usage.

Job schedule will be used to complete the scenario of services reservation, and then manage the load. SOM technique is also used to do the middleware self-management process. Other types of machine learning (like SVM [16, 17]) will be test in the future to do the classification process. For the future, this environment will be developed to be as a web services that can be added to OGSA middleware services.

The idea of autonomic monitor will be develop in the future stages to predict the type of data that is needed by the autonomic services. Also, it will be used to reduce the amount of unnecessary data that is transfer from the target to the log file system

References

1. V. Berstis, Fundamentals of Grid Computing, 2002.
2. M. Y. Chen, E. Kıcıman, E. Fratkin, A. Fox, and E. Brewer, Pinpoint: Problem Determination in Large, Dynamic Internet Services, 2002.
3. M. LaMonica. IBM draws self-management blueprint, April 2003.
4. M. Chen, A. X. Zheng, J. Lloyd, M. I. Jordan and E. Brewer, Failure Diagnosis Using Decision Trees. 2004.
5. G. Candea, M. Delgado, M. Chen and A. Fox, Automatic Failure-Path Inference: A Generic Introspection Technique for Internet Applications, June 2003.
6. M. Y. Chen, A. Accardi, E. Kıcıman, J. Lloyd, D. Patterson, A. Fox and E. Brewer, Path-Based Failure and Evolution Management, 2003
7. U. Heuser, J. Goppert, W. Rosenstiel and A. Stevens, Classification of Human Brain Waves using Self-Organizing Maps, August 1996
8. N. N. Schraudolph and T. J. Sejnowski, Competitive Anti-Hebbian Learning of Invariants, 1992.
9. R. H. White, Competitive Hebbian Learning 2: an Introduction, 1992.
10. J. Vesanto, Using SOM in Data Mining. Licentiate's thesis. Thesis for the degree of Licentiate of Science in Technology, Supervisors: Professor Olli Simula, Professor Samuel Kaski, Espoo, Finland 17th April 2000.
11. R. Pollock, T. Lane and M. Watts, A Kohonen Self-Organizing Map for the functional classification of proteins based on one-dimensional sequence information, 2001.
12. E. Bingham, J. Kuusisto and K. Lagus, ICA and SOM in Text Document Analysis, August, 2002.
13. Haynos, J.U.a.M., A visual tour of Open Grid Services Architecture, August 2003.
14. J. Vesanto, J. Himberg, E. Alhoniemi and J. Parhankangas, Self-Organizing Map in Matlab: the SOM Toolbox. In proceedings of the Matlab DSP Conference 1999, Espoo, Finland, pp. 35-40, 1999.
15. T. Joachims, Text Categorization with Support Vector Machine: Learning With Many Relevant Features. ECML-98. 10th European Conference on Machine Learning, Heidelberg, Germany, 1998.
16. C. Hsu, C. Chang, and C. Lin, A Practical Guide to Support Vector Classification, 2003.
17. D. Arnaud, Study of a document classification framework using Self-Organizing Maps, October 1, 2003
18. T. Kohonen, The Self-Organizing Map (SOM). October, 2000. http://www.cis.hut.fi/projects/somtoolbox/theory/somalgorithm.shtml
19. http://www.cs.toronto.edu/~delve/methods/knn-class-1/knn-class-1.html