COLLABORATIVE FILTERING TO GUIDE THE USE OF WEB BROWSERS

D A Sanders, J Bergasa-Suso, S Chester, G E Tewkesbury and J Graham-Jones

Systems Engineering Research Group, University of Portsmouth, Portsmouth, PO1 3DJ, UK.

Email: *david.sanders@port.ac.uk*

Abstract

A lack of filtering to provide access control and the complexity of web-based systems makes it difficult to use the internet, especially in virtual teams. This paper identifies a need for new flexible and adaptable filtering methods and describes a model of a filtering method. The systems described will overcome the limitations of current methods and help to provide structured, focused and controlled access to the Internet. A first simple system called iLessons is described that is embedded within Microsoft Internet Explorer 6 and provides tools to: create web pages; define zones of the Internet that can be accessed by virtual teams and enforce settings in a set of computers.

Keywords: WWW, browser, filter, internet, virtual teams.

Introduction

New tools were required to provide structured, focused and controlled access to the Internet during work in virtual teams. There can be a lack of control over pages that team-members access and systems can be complex to manage [1].

This paper describes a filtering method using a model that can be combined and extended. A new system is developed from these models that attempts to overcome some of the limitations of current systems. The system is called iLessons and is a web browser-based system embedded within Microsoft Internet Explorer 6. The use of a standard web browser makes the system easy-to-use. Focus is provided by allowing team leaders to decide which sort of Internet pages or domains team members are allowed to access; structure is provided by WYSIWYG HTML page editing within the browser, as well as the ability to collect and reuse Internet resources by drag & drop; control is achieved by saving this information into a single lesson file that can be applied to computers grouped by teams or tasks.

A number of non-intelligent web page filtering methods were considered for this system but Artificial Intelligence algorithms provided a more reliable and flexible filtering. By using intelligent filtering methods, the set of pages that were accessable was not limited by URLs or keywords, but by the relevance of the page contents to a task or subject being explored. The use of intelligent filtering methods to allow users to access any page related to a subject created the need for new tools to assist team members in sharing knowledge. These new tools were also needed to adapt to the different "learning styles" of different users, to improve effectiveness.

There was a broad range of web server-based learning systems in the market, such as the Blackboard Learning System [2], WBT Systems Top Class [3], WebCT Campus Edition [4] and the like. All these systems provided web-based tools to: manage internet access, create material, allow users to access the internet and collaborate remotely. These tools were stored and run from a web server that could be accessed from any web browser. They did not provide any content-specific filtering of web pages, so users could lose concentration and navigate to unrelated web sites; activity which these systems could not detect and react to.

This navigation problem has been solved in this work by using a proxy server in conjunction with web server-based systems. The proxy server sat between a client application such as a web browser and another server such as a web server, as described by Webopedia [5]. The proxy server improved overall network performance by caching information retrieved from other servers, but most importantly, it filtered requests

sent to the Internet.

The use of server-based systems has involved the need to set up and maintain a network infrastructure. Also only a limited functionality was offered from the available clients, as most tools were web-based and run from the server side. These limitations uncovered the need for another type of software tool that could be used with minimal network infrastructure, such as a single computer with a dial-up connection, and that provided extended functionality through the use of client-based, user-centred intelligent tools to assist students in their learning experience.

Client-based systems such as IBM Lotus Virtual Classroom [6] provided extended functionality such as whiteboard, screen sharing, video conferencing and automatic assessment, but still relied on a proprietary server platform and were not easy to use.

Client-based systems that do not rely on proprietary servers had the advantage that no extra network infrastructure was needed to run them; there was no need for a proxy server or a web server. Files could be stored in a network file server or locally in the case of single machines. This also meant that these systems were scalable; a single client could be installed in a computer with a dial-up connection or in every computer in a school network without extra overhead.

A benefit of web sever-based systems was that they could be accessed from a standard web browser, which made them easy to use. When using a proprietary client system, users needed to learn how to use the new system. Reusing current functionality available in commercial web browsers made the system easier to use and to implement. There were a number of good quality web browsers in the market, such as NCSA Mosaic [7], Netscape Navigator [8] and Microsoft Internet Explorer [9]. Internet Explorer's functionality could be expanded by explorer bars, tool bands, Browser Helper Objects (BHOs) and Asynchronous Pluggable Protocols (APPs). This technology was used in this research to provide extended functionality within a standard web browser.

Internet Explorer 4 introduced Explorer Bars to display information and interact with the user [10]. BHOs were COM components that were loaded by Internet Explorer each time it started [11]. Such objects could perform any action on the available windows and modules. For example, a BHO could detect the browser's events; access the browser's menu and toolbar; create windows to display additional information on the currently viewed page, and install hooks to monitor messages and actions. APPs were COM components that were loaded by Internet Explorer to handle a URL request depending on its protocol scheme, such as "http:" "ftp:" or other custom protocol schemes.

ILessons

ILessons was a product of this research and was written in Visual C++. ILessons was based on the Internet Explorer extensions described above and is available for viewing and testing at *http://www.ilessons.co.uk*. The authors would welcome questions and comments.

The user must specify the location of the computer where the system is installed when an iLessons system is set up. The system can then control a set of computers grouped by virtual team. All the information necessary for iLessons to work is located in a shared network folder or in a local folder if the system is run in a single computer. This includes encrypted user login information, the location of each computer and the work to be undertaken.

Team Leaders can collect resources from the Internet by dragging and dropping them into the left-side pane of the iLessons system. Resources are grouped into plain text, images, links and html blocks and can be used later to create lesson web pages. Resource collections can be saved into a single file and shared with other users. Team Leaders can create a set of lesson web pages by turning Internet Explorer into a web page editor. Resource collections can be loaded and resources dragged and dropped back to the lesson page. Custom content and formatting can also be applied, including tables, lists and layers. All the web pages of a lesson can be saved into a single file and shared with other users or assigned to a particular classroom or set of computers to be used by students. Team Leaders are recommended to keep links to all the resources that are used in their lesson web pages and add them as references in order to comply with copyright laws.

Apart from creating lesson web pages, Team Leaders can also allow or deny specific areas of the Internet so that students have limited but focused access to the Internet. Web pages can also be "trusted", so that the Team Leader sets a certain depth factor. When students are viewing a trusted page, they are automatically allowed to navigate to any link from the page and to any link from those pages, until the depth specified by the Team Leader is reached. Team Leaders can assign a lesson file to a particular classroom. When this occurs, all the computers registered in that classroom automatically load the lesson and Internet access is restricted by the navigation zone specified within the lesson. As the lesson files are stored in a network location, the Team Leader can log off or even switch off the computer without affecting the student computers. When no lesson is being implemented, the student clients load a default lesson. A different default lesson can be specified for each classroom and may just be a "total-block" page or a departmental page with links to internet resources.

When the system is being used in student mode, it automatically displays the main page of the lesson being implemented and team members are able to navigate to pages created by the Team Leader. When students request web pages then only those belonging to an allowed or trusted zone of the Internet are allowed. The student is also able to create coursework files in Word or Excel format from within Internet Explorer.

URL and full text filtering

A URL database containing allowed, denied and trusted URLs and URL fragments was populated by a Team Leader during the creation of a collaborative task, using a database engine integrated within the system. When a page request from a team member was received by the URL filter during a task then permission was sought by the database engine using the page's URL. If permission was granted access to the page was granted accordingly. Otherwise, default permission was applied.

Full text filtering

Full text filtering consisted of searching each page for a set of keywords. A Team Leader populated a keyword database using a database engine integrated within the system. When the text filter received page requests from students during a lesson, each word in the page was sought in the keyword database using a database engine. Access to web pages was determined by the text filter depending on the presence of keywords in the page.

Team Leaders populated the keyword database with a set of subject-related keywords that must appear in the pages requested, and a set of keywords, which prevented pages from being displayed.

URL and full text filtering

URL filtering and full text filtering were combined to make a more flexible filter. A Team Leader populated a URL database and a keyword database during the creation of a lesson, using a database engine integrated within the system. When the URL and text filter received page requests from students during a lesson, the page's permission in the URL database was sought by the database engine using the page's URL. If permission was found, access to the page was granted accordingly. Otherwise, each word in the page was sought in the keyword database using a database engine. Access to web pages was determined by the text filter depending on the presence of keywords in the page.

URL filtering was selected for iLessons because it was a fast, simple and accurate filtering method. Team Leaders could easily allow, deny and trust access to a set of pages related to the subject, and the URL database was stored in the lesson file. Systems used by the team members read the lesson file and access to the Internet was restricted to areas specified by the Team Leader.

ILessons was an easy-to use, novel and effective tool to deliver guided learning using the Internet in a classroom but it lacked flexibility, adaptability and collaboration facilities to enable teams to use the Internet as a research group tool. The research work moved on to the creation of a new intelligent system that would provide these functions, using iLessons as a platform.

Supervised Machine Learning

In order to provide more reliable and flexible filtering, Supervised Machine Learning (SML) algorithms discussed and compared by Yang *et al* [12] were considered. By using SML – based filtering methods, the set of pages that students were able to access was not limited by a group of URLs or keywords, but depended on the relevance of the page contents.

Supervised Machine Learning involved previous training before using a system in order to determine a relevance pattern to identify which pages a user could access. Users trained the system by providing a set of relevant and irrelevant web pages for a particular subject. A document analysis engine analysed the text contained within a page and a pattern finding engine obtained a relevance pattern from the training set. The relevance pattern was loaded into the student system during a lesson. When team members requested a web page, the document analysis engine extracted the contents, which were compared by the pattern finding engine with the relevance pattern obtained from the training set and filtered by a relevance filter; access was granted to pages that matched the subject's relevance pattern within a threshold.

Subject-specific filtering using document categorisation was flexible enough to enable students to use the Internet as a research tool while keeping them focused on a subject.

Algorithms such as Support Vector Machine (SVM), k – nearest neighbour (kNN), Linear Least Squares Fit (LLSF) and Naive Bayes probabilistic classifier (NB) or Artificial Neuronal Networks (ANN) could be used to automatically extract semantic features from documents. Results of the comparison of these classification methods running tests between method pairs indicated that k-nearest neighbour was the most accurate [12]. About 20 documents had to be rated in each category to reach 60% of accuracy.

A drawback of using supervised machine learning algorithms was that a pattern had to be applied each

time that a document was retrieved in order to assess its suitability. URL filtering was combined with SML to cache the relevance rating of each document in a URL database. Team Leaders trained the system by providing a set of relevant and irrelevant web pages for a subject. A document analysis engine analysed the text contained within a page and a pattern finding engine obtained a relevance pattern from the training set. The relevance rating was found by a relevance filter and indexed in a URL database using the URL of a document in a database engine. The relevance pattern was loaded and when team members requested a web page, the URL was extracted by the document analysis engine and sought in the URL database by the database engine. If no permission was found, the document analysis engine extracted the contents, which were compared by the pattern finding engine with the relevance pattern obtained from the training set and filtered by a relevance filter. Access to pages that matched the subject's relevance pattern within a threshold was granted and the database engine cached the document relevance in the URL database, to be used in subsequent requests, so that each document was categorised only once.

Agent-mediated document filtering

SML – based filtering methods provided effective and flexible filtering of web pages, allowing users to research using the Internet while keeping them focused on a subject. New filtering methods were necessary to provide assistance to users as well as focus. Semantic networks and intelligent agents were combined with the filtering methods described above to create new intelligent filtering methods that also assisted the user in researching using the Internet.

Autonomous agents were utilised to analyse and react to user behaviour. Patterns found by agents could be loaded into a filtering system or utilised to assist users by recommending web sites or assessing the relevance of a page. Agents also performed pre-emptive analysis of pages linked from a page, or used network idle time to conduct searches based on a certain interest pattern found in a user's behaviour. Agents did not need to be trained and they made search and navigation through documents more effective, although the learning curve was slow. Peña *et al* [14] [15] describe the use of agents to filter web pages.

Agent-mediated collaborative document filtering involved a group of user agents communicating between themselves. User agents compared user behaviour and document classification patterns in order to improve overall group performance. Green *et al* [16] and Papaspyrou *et al* [17] describe such systems. The system produced was Machine learning with file name / URL filtering and collaborative agents.

Supervised Machine Learning enhanced with URL filtering was combined with a system of collaborative agents. Each user had an agent assigned. A user agent monitored each user. The agent broadcasted new URL database entries and updated the user's URL database with received entries. In this way, each document was classified only once within the workgroup.

Supervised Machine Learning combined with Semantic Networks was extended with a system of collaborative agents. A user agent monitored each user. The agent broadcasted new semantic network database entries to other agents and updated the user's semantic network with received entries. Agents assisted users by suggesting related web sites found by other users.

Devedžić [18] stated that "Next-generation web-based applications should exhibit more theory and content-oriented intelligence and adaptability, pay more attention to interoperability, reusability, and knowledge sharing issues, and look more closely to general trends in web development".

In order to make WWW Page content adaptable, reusable, shareable and interoperable between systems, it had to be machine-readable, so agents and other software entities could use it. This was achieved by structuring the content using ontologies. In this paper, ontologies are described as content theories about the sorts of objects, properties of objects, and relationships between objects that are possible in a specified domain of knowledge [19]. Material was organised in ontologies, such as "subject", "introduction", "description" or "questions". Ontologies could also link to other documents.

Using ontology-based content, filtering was performed based on the relevance of a web page or other document determined by the ontologies that they contained and their content. Content could be reused and supporting content could be retrieved by agents.

Team Leaders constructed lessons by reusing available ontology-based material found by an ontology search agent using an ontology-based edition subsystem. Web pages with no ontologies were analysed by a document analysis engine that divided the contents into ontologies. An ontology profiler analysed the ontologies being used by the Team Leader and their contents, creating an ontology profile of which ontologies and contents were suitable for the lesson.

During a lesson, team members accessed the ontology-based material, as well as other material. If the WWW based material was not ontology-based, a document analysis engine inferred ontologies. An ontology-based filter allowed only suitable content by using an ontology profiler and the profile created by the Team Leader. An ontology-teaching agent monitored user actions and assisted the student by suggesting related WWW Page content, using an ontology search agent.

The system was improved using collaborative agents to allow users to automatically assist each other while researching on a common subject using the Internet. Teaching agents suggested new relevant content to other users by using a collaboration agent. This agent also received information about relevant content from other users, which was suggested by the teaching agent, enhancing the group learning experience.

Discussion

iLessons was a product of this research created to provide easy-to-use ways to control and focus Internet usage to a subject, to create custom web-based content and to collect and reuse Internet resources such as text or images. Current filtering methods were considered and new models of these systems were created to compare the systems in terms of accuracy, complexity and ease of use. URL filtering proved to be an accurate, simple and easy to use method and was selected for iLessons.

URL filtering prevented the system from being used as a research tool that would allow students to freely navigate to any web page related to a subject. A flexible filtering method based on the relevance of the contents to a subject was necessary. Two new models of intelligent filtering methods were created based on supervised machine learning algorithms.

The intelligent filtering methods were flexible and allowed researching within a subject using the Internet, but they did not assist users or allow team members to share discovered pages.

New intelligent filtering models and methods were created to overcome these limitations by combining the intelligent filtering methods with semantic networks and intelligent agents.

Conclusions

The systems are to assist team leaders by releasing them from monitoring the use that workers make of the Internet and making them more available.

iLessons proved to be an easy-to-use and effective new system that assists team leaders in using the Internet as a guided tool by providing a means of focusing, structuring and controlling use of the WWW.

Effective Internet systems must allow users to navigate to web pages related to a subject, rather than limiting the use of the WWW to a predefined set of pages. Effective Internet systems must also assist the user and allow sharing of discovered information between users.

The system appears to improve effectiveness and efficiency of group working.

References

[1] J. Bergasa-Suso, D. A. Sanders, A. Close, and G. E. Tewkesbury, "A caught in the act filter to assist in using the Internet". *Proceedings of 4th International Conference on Information Comm's Technologies in Education*, 2003, pp. 225-230.

[2] Blackboard Inc. (2004, Jul.) Blackboard Learning System. [Online]: http://www.blackboard.com/products/academic/ls/index

[3] WBT Systems (2004, Jul.) e-Learning from WBT: TopClass LCMS. [Online]: www.wbtsystems.com

[4] WebCT (2004, Jul.) WebCT Campus Edition. [Online]: http://www.webct.com/software

[5] Webopedia (2003, Oct.) Proxy server. [Online]: http://www.webopedia.com/TERM/p

[6] IBM Corporation (2004, Jul.) IBM Lotus Virtual Classroom. [Online]: http://www.lotus.com/lotus/offering7.nfs/wdocs/

[7] National Center for Supercomputing Applications (2001, Dec.) NCSA Mosaic. [Online] Available: http://archive.ncsa.uiuc.edu/SDG/Software/Mosaic/NCSAMosaicHome.html

[8] Netscape Corporation (2001, Dec.) Introduction: Netscape 6.1 at glance, Highlights of Netscape 6.1. [Online]: http://home.netscape.com/browsers/6/revguide

[9] Microsoft Corporation (2001, Dec.) Microsoft Internet Explorer 6: Technology Overview. [Online]: http://www.microsoft.com/windows/ie/evaluation

[10] Microsoft Corporation (2001, Dec.) Creating Custom Explorer Bars, Tool Bands, and Desk Bands [Online]: http://msdn.microsoft.com/library/default.asp?url=/

[11] D. Esposito (2004, Jul.) Browser Helper Objects: The Browser the Way You Want It [Online]: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwebgen/html/bho.asp

[12] Y. Yang, and X. Liu, "A re-examination of text categorization methods" *Proceedings {SIGIR}-99, 22nd {ACM} International Conference on Research and Development in Information Retrieval*, 1999, pp. 42 – 49.

[13] C. I. Peña, J. L. Marzo, and J. L. De la Rosa, (2002), "Intelligent Agents in a Teaching and Learning Environment on the Web", In Proceedings of the *2nd IEEE Int' Conf' on Advanced Learning Technologies (ICALT2002)*, pp. 21 – 27.

[14] C. I. Peña, J. L. Marzo, and J. L. De la Rosa, (2002), "Student modeling using intelligent agents in a web-based teaching and learning" [Online]: http://eia.udg.es/~atm/bcds/pdf/aia2002-mayo6-udg.pdf.

[15] S. Green, C. Padraig, and S. Fergal, "Agent Mediated Collaborative Web Page Filtering" In *Cooperative Information Agents II, Learning, Mobility and Electronic Commerce for Information Discovery on the Internet, 2nd Int' Workshop, CIA' 98, Paris, France, July 4-7, 1998*, pp. 195 – 205.

[16] N Papaspyrou, C Sgouropoulou & E Skordalakis, "*A Model of Collaborating Agents for Content-Based Electronic Document Filtering*", Jnl of Intelligent & Robotic Systems, 26(2), pp. 199 – 213, 1999.

[17] V. B. Devedžić, "*Key issues in Next-Generation Web-Based Education*", IEEE Transactions on Systems, Man and Cybernetics – Part C: Applications and Reviews, 33(3), pp. 339 – 349, 2003.

[18] B. Chandrasekaran, J. R. Josephron, and V. R. Benjamins, "*What areontologies, and why do we need them?*", IEEE Intelligent Systems, 14, pp.20-26, Jan – Feb 1999.

[19] K.A. Papanikolaou, M. Grigoriadou, G.D. Magoulas and H. Kornilakis, "*Towards New Forms of Knowledge Communication: the Adaptive Dimension of a Web-based Learning Environment*", Computers and Education, 39(4), pp. 333-360, 2002.