A Consideration of Query Improvement to the SMOKA Astronomical Archive System

Yilang Wu
University of Aizu
Aizu-wakamatsu
Fukushima, Japan, 965-0085
d8152103@u-aizu.ac.jp

Wanming Chu
University of Aizu
Aizu-wakamatsu
Fukushima, Japan, 965-0085
w-chu@u-aizu.ac.jp

Subhash Bhalla
University of Aizu
Aizu-wakamatsu
Fukushima, Japan, 965-0085
bhalla@u-aizu.ac.jp

ABSTRACT

The Web technology has brought the significant changes to astronomical archive services. In this study, we investigate the limitations in data search and query of the SMOKA (Subaru-Mitaka-Okayama-Kiso-Archive) astronomical archive system, which heavily depends on Java Server Pages (JSP) and Servlet web technology. By discussing the new trends of the software systems for astronomy, we propose to utilize the emerging or popular web technologies and also the novel query interfaces to improve the query service in SMOKA.

Keywords

Astronomical Data Archive, SMOKA, Query, Web Interface

1. INTRODUCTION

The Subaru-Mitaka-Okayama-Kiso-Archive (SMOKA) is a public science archive system which provides access to the data of the Subaru Telescope, the 188 cm telescope at Okayama Astrophysical Observatory, and the 105 cm Schmidt telescope at Kiso Observatory/University of Tokyo [1]. The normalization of Flexible Image Transport System (FITS) keywords among various instruments of the SMOKA associated Telescope makes it easy to construct and maintain the database. As a successor of the MOKA3 [2], the SMOKA is mainly constructed using Web technology such as Java Serverletand JavaServer Pages,which is fit to the component-based development for the multi-tiered system architecture. The UI-tier of SMOKA provides a well-defined interface to access the data. The middle-tier is in charge of retrieving the data from the back-end database-tier. Such component-based approach and multi-tiered architecture derived a number of separated search functions (see Table 1).

However, such system framework and architecture is insufficient for query needs and further development nowadays, because of its separated search interfaces and duplicated search functions. The domain specific users [3] such as amateur or professional astronomers prefer 1) easily understandable user interface, for instance with the style of QBO (Query-By-Object) [4]; 2) facilitating work-flow between query steps, such as the support of multi stage query [5]; 3) reusing or adapting existing services or interfaces as much as possible, and eliminating the duplicated features.

Table 1: SMOKA Search Functions and Interfaces

Search Types	Archives					Mixed With	Search Functions
	S	O	K	M	H		
Simple Search	○	○	○	○	○	—	Search data from a list of object names
Advanced Search	○	○	○	○	○	SUP, KWF	Data from various search constraints.
SUP Search	○	○	○	○	○	—	Search raw data and astrometric calibrated data.
KCD (pin-point) search (KCD)	○	○	○	○	○	—	Search frames whose fields contain the search coordinates.
Calendar Search	○	○	○	○	○	—	Search data from a calendar (observation date) with weather data.
MB Search	○	○	○	○	○	SUP, KCC, KCD	Search data which may contain minor bodies in the solar system.
All Keywords Search	○	○	×	×	○	—	Search data by FITS Header Keywords values.
Full-Text Search	○	○	○	○	○	—	Search any words in FITS headers (including History and Comment).
Overlapped Area Search (Old & New)	○	○	○	○	○	SUP, KCD	Search areas(s) observed multiple times.
Summit Log Search	○	×	×	×	×	—	Search any words in Summit Logs.
Number of Frame Search	○	○	○	○	○	—	Search number of frames and IDs from object coordinates.
Description	S (Subaru), O (OAO), K (Kiso), M (MITSuME), H (HHO) ○ (Supported), × (Not Supported)						

In this study, we investigate the system limitations in SMOKA and analyze the prominent software systems which have been widely used in recent popular astronomical data archives, and propose to utilize the emerging web technology and interface to improve the query service in SMOKA.

2. BACKGROUND

On the top of the emerging web technologies, the once monolithic and file-server oriented web servers are evolving into easily programmable server applications capable to cope with the complex interactions made by new generation of browsers [6]. Accessing astronomical data archives, schedule, control and monitor observatories, and in particular robotic telescopes, supervising data reduction pipelines, all are capabilities that can now be implemented in a JavaScript web application.

The emerging high-end web technology with open and well documented interface is considered to be the sustainable approach to further advance many scientific and astronomical data resources and systems. It is a trend that many astronomical data archives are merging together into larger data repository with more advanced search and query interface.

IWAIT '15, Oct. 8 – 10, 2015, Aizu-Wakamatsu, Japan.

The CDS portal[1] is such a mash-up application of various web based searching services for astronomical archive data. It aims at providing a single entry point to search and access the different CDS services (such as the Simbad VizieR, Aladin, X-Match) and facilitating the work-flow between the services. Nevertheless, it opens rich programming interfaces for developers, including various Java libraries, Unix/Linux clients, XML Web services, jQuery Widget scripts; furthermore, it also targets the smartphone OS (Android and iOS), HTML5 and WebGL developers. Various sources code of CDS Portal components is available on public (for instance via Github[2]) with detailed technical documents, and it has formed a large open source communities, connecting with the world wide users and developers.

3. DISCUSSION

The JSP, which is used by SMOKA is better to be treated only as a presentation (or representation) technology, not as the central structuring technology of the application. The problems of developing a pure JSP web-app without framework are endless; for instance there are limited abilities to usefully re-use code across pages, difficult error handling, limited ability to implemented caching strategies, tedious validation logic, and constant battle to ensure HTTP GETs. Thus the SMOKA adopted the JSP together with Servlet to overcome that limitations, and provide a component-based, platform-independent method for building Web-based applications, without the performance limitations of Common Gateway Interface (CGI)[3] programs. The Servlets is advanced in the following features: 1) portability across operating systems and across web servers; 2) harnessing the full power of the core Java APIs, such as networking and URL access, multi-threading, image manipulation, data compression, JDBC, object serialization, internationalization; 3) efficiency and endurance in memory manipulation; 4) supporting safe programming since it inherits Java's strong type safety, exception-handling mechanism; 5) elegant with clean code and models; 6) tightly integrated with the server. However the design in Servlet is difficult and slows down the application; developers have to write complex business logic, making the application difficult to understand; the maintenance of Java Runtime Environment on sever side results in extra work to administrators.

There is increasing demand of easy to learn and start with framework and technology to build data-intensive real-time applications running across distributed devices. Along with the development of HTML and HTTP, JavaScript, the browser-side programming language has become a lot more powerful. The Node.js[4] is such popular technology, with coders everywhere using it to create APIs and build a new matrix of interoperability across the Internet. Furthermore, according to the experiment in [7], Node.js performs much better than the traditional technology such as PHP in high concurrency situation, no matter in benchmark tests or scenario tests. PHP handles small requests well, but struggles with large requests. Besides, Node.js prefers to be used in the IO-intensive situation, not compute-intensive sites; but that is enough to support the query to the astronomical big data. On the other hand Python-Web is also not suitable for the compute-intensive website.

4. PROPOSAL

A series of novel web technologies, such as Node.js with C++ *add-ons*, SVG[5] and HTML5 web based visualizing tools, MongoDB for complex data structures have been proposed for astronomy in paper [6]. Based on the aforementioned technologies, we propose to upgrade the software system in SMOKA by integrating the the existing but independent search functions (see Figure 1) into a simple query interface such as the QBO and multi stage query [5], in which users can make complex query with guidance in steps. And to fully utilize the advanced new web technologies, the open and well-documented programming interface for data access is highly suggested to connect the world wide developers.

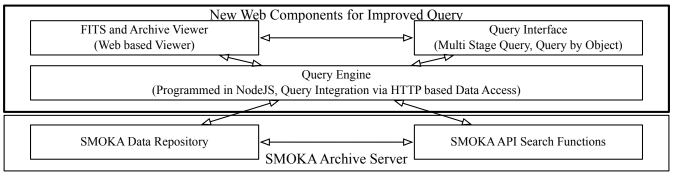
New Web Components for Improved Query
FITS and Archive Viewer
(Web based Viewer)
Query Interface
(Multi Stage Query, Query by Object)
Query Engine
(Programmed in NodeJS, Query Integration via HTTP based Data Access)
SMOKA Data Repository
SMOKA API Search Functions
SMOKA Archive Server

Figure 1: Proposed System Architecture

5. SUMMARY AND FUTURE WORK

To ease the system development and maintenance and to improve user experience for SMOKA, it is time to embrace the new generation of Web technology and more open community. We plan to build new query application by introducing the simple query interface like QBO, multi-stage query work-flow, and also eliminating the duplicated functions.

6. REFERENCES

[1] H. Baba, N. Yasuda, S.-I. Ichikawa, M. Yagi, N. Iwamoto, T. Takata, T. Horaguchi, M. Taga, M. Watanabe, T. Ozawa, and M. Hamabe, "Development of the Subaru-Mitaka-Okayama-Kiso Archive System," *Adass Xi*, vol. 281, pp. 298–+, 2002.

[2] T. Horaguchi, E. Nishihara, M. Yoshida, K. Aoki, T. Ito, M. Watanabe, S. Ichikawa, T. Takata, S. Yoshida, and M. Hamabe, "An astronomical data archive system with a java-based user interface," *Publications of the Astronomical Society of Japan*, vol. 51, no. 5, pp. 693–701, 1999.

[3] A. Madaan and W. Chu, "Handling domain specific document repositories for application of query languages," in *Databases in Networked Information Systems* (A. Madaan, S. Kikuchi, and S. Bhalla, eds.), vol. 8381 of *Lecture Notes in Computer Science*, pp. 152–167, Springer International Publishing, 2014.

[4] A. Yasir, M. Kumara Swamy, P. Krishna Reddy, and S. Bhalla, "Enhanced query-by-object approach for information requirement elicitation in large databases," in *Big Data Analytics* (S. Srinivasa and V. Bhatnagar, eds.), vol. 7678 of *Lecture Notes in Computer Science*, pp. 26–41, Springer Berlin Heidelberg, 2012.

[5] Y. Wu and W. Chu, "Query languages for domain specific information from ptf astronomical repository," in *Databases in Networked Information Systems*, pp. 237–243, Springer, 2015.

[6] P. Sprimont, D. Ricci, and L. Nicastro, "New web technologies for astronomy," in *Revista Mexicana de Astronomia y Astrofisica Conference Series*, vol. 45, p. 75, 2014.

[7] K. Lei, Y. Ma, and Z. Tan, "Performance comparison and evaluation of web development technologies in php, python, and node. js," in *Computational Science and Engineering (CSE), 2014 IEEE 17th International Conference on*, pp. 661–668, IEEE, 2014.

[1] http://cdsportal.u-strasbg.fr

[2] http://github.com

[3] The Common Gateway Interface (CGI) is a standard for interfacing external applications with information servers, such as HTTP or Web servers.

[4] http://nodejs.org

[5] Scalable Vector Graphics