Engineering the ATLAS TAG Browser

Qizhi Zhang for the ATLAS Collaboration
Argonne National Laboratory, High Energy Physics Division, Building 360, 9700 S.
Cass Avenue, Argonne, IL 60439, USA
E-mail: qzhang@anl.gov

Abstract. ELSSI is a web-based event metadata (TAG) browser and event-level selection
service for ATLAS. In this paper, we describe some of the challenges encountered in the
process of developing ELSSI, and the software engineering strategies adopted to address those
challenges. Approaches to management of access to data, browsing, data rendering, query
building, query validation, execution, connection management, and communication with
auxiliary services are discussed. We also describe strategies for dealing with data that may
vary over time, such as run-dependent trigger decision decoding. Along with examples, we
illustrate how programming techniques in multiple languages (PHP, JAVASCRIPT, XML,
AJAX, and PL/SQL) have been blended to achieve the required results. Finally, we evaluate
features of the ELSSI service in terms of functionality, scalability, and performance.

1. Introduction
The ATLAS (A Toroidal LHC Apparatus) TAGs are event-by-event metadata records containing
about 280 key quantities that identify and describe the event and sufficient navigational information to
allow access to the event data at all prior processing stages. These attributes are decided by ATLAS
2006 Task force and maintained by the Physics Analysis Tools group. The TAG database
encompasses file- and relational-database-resident event-level metadata, distributed across all ATLAS
Tiers[1].

ELSSI is a web-based TAG browser and event-level selection service for ATLAS [2]. It was
initially built as a browser of TAGs hosted on a server machine at CERN (European Organization for
Nuclear Research). While TAG datasets have gone through significant content evolvement starting
from computing system testing and detector commissioning activities in 2008 through to real and
simulated data taken in 2008, 2009 and 2010, ELSSI has been cloned to serve each specific TAG
dataset[3].

Over time, two other databases have been developed which are used by ELSSI. The TASK (TAG
Application and Service Knowledge) database[4] tracks the status and location of uploaded TAG data.
The COMA [COditions MetadatA)][5] contains run level metadata and trigger information which is
needed for forming meaningful TAG queries. Both TASK and COMA have expanded in both schema
and data volume.

As ATLAS collects data and users develop their physics analysis, more TAGs have been produced.
TAG datasets have been routinely uploaded into Oracle databases and distributed across continents.
The hosting topology of TAG datasets becomes heterogeneous[6]. With the overall goal of load-
balancing the TAG infrastructure, Oracle databases at CERN and several Tier-1 and Tier-2 sites
(BNL, TRIUMF, DESY, PIC, and more to come) are hosting different sets of TAG data with or without overlaps at the run level.

In addition, the underlying TAG related services such as Extract, Skim, Trigger Decoding, runBrowserReport and runBrowser have emerged and substantially matured[7-8]. User feedback from tutorials, software week and regular analysis practice has increased and served as major source of new feature and improvement requirements to the ELSSI development.

Given the above facts, the need for an integrated interactive event level selection service, iELSSI, for all foreseeable scenarios—whatever, whenever, wherever and however TAG datasets are distributed—is inevitable. In the following sections, we report the challenges encountered in the process of developing iELSSI, and the software engineering strategies adopted to address these challenges.

2. Discover

2.1. Integrated ELSSI

Initially, changes to the data model were handled by deploying a new version of ELSSI, which begat a proliferation of multiple slightly mutated versions of ELSSI. This model finally broke when it was decided to host Monte Carlo (MC) TAGs only at DESY instead of CERN. There was an immediate need to integrate all TAG data sources under one single ELSSI service, no matter where the dataset would be hosted.

When ATLAS collected its first batches of collision events, TAG data for the collision have been produced and uploaded. Therefore, ELSSI needs to accommodate not only the previous TAG datasets, but also the physics TAGs, their re-processing results and any other emerging TAGs at all possible hosting sites.

The architecture redesign was made possible by the completion of the TASK database. This database provides information about the uploaded TAGs, including (but not limited to) details of data projects, datasets, data collections, runs, schemas, sites, AMI (ATLAS metadata interface) tag and connection settings.

2.2. More challenges

Starting May, 2010 it was realized that the same host site had more than one database schema that housed the same TAG “project” but with different “processing” or “type”. For instance, project “data10” is stored under four schemas at PIC:

| PROJECT | SITE | SCHEMA_NAME | TYPE   |
|---------|------|-------------|--------|
| data10  | PIC  | ATLAS_TAGS_COMM_F_2010 | TAG_COMM |
| data10  | PIC  | ATLAS_TAGS_COMM_R_2010 | TAG_COMM |
| data10  | PIC  | ATLAS_TAGS_DATA_F_2010 | TAG     |
| data10  | PIC  | ATLAS_TAGS_DATA_R_2010 | TAG     |

Up to this point ELSSI had been designed and implemented under the assumption that all datasets at a given site was under the same schema. This assumption was broken when a user selected two data09 “collections” with one of them in the first pass and the other one in the December reprocessing. ELSSI received error from Oracle telling ‘table not exist’.

Exactly how the TAG datasets are divided and stored under Oracle has a significant effect on the ELSSI development. All the information displayed for selections under “Stream”, “Triggers” and “Physics attributes” is “project” specific. Now that the “project” is not unique enough, everything else should be narrowed down to the schema level too. The ELSSI Versions 3.x were released to take care of the multiple schemas under one “project”.

2
2.3. Site transparent iELSSI
While more TAG data have been uploaded to different hosting sites, the topology of TAG data distribution also extends. Through one user’s analysis requirement, we have discovered a new scenario: The user wanted to select events for runs 155073 and 155160. He noticed that data collection “data10_7TeV_physics_MuonswBeam_f255_m471_READ” was being hosted at two sites: CERN and PIC, but the run 155073 was missing from the CERN site. As a result, the user would have to select CERN to retrieve event data for run 155160 and then start a new event selection window for retrieving run 155073 from site PIC. This is not only inefficient but also lacking the ability to combine the retrieved data from these two runs. Or if the user had spent time browsing the ELSSI page ahead of time to know that PIC had both runs of his interest, he could directly select site PIC and perform the event selection from one site only. Either way, requiring the user to know about the data distribution among database hosting site is not a user-friendly strategy. Therefore, a site transparent event selection service is required. Meanwhile, other TAG related services have emerged and, together with ELSSI, are managed under the umbrella called “ELSSI Suite”. To distinguish the web-based event selection service from the others, it will be renamed to “iELSSI” starting from the site transparent version.

Figure 1. The parent layer provides summary info about TAG datasets and forks ELSSI child services according to user’s selection of TAG dataset(s) and run(s).
3. Design

3.1. Parent layer
According to the design specification, the integrated ELSSI should start with a page which at first only presents the list of names of datasets whose TAGS have been uploaded into the Oracle database. The user can then select one of the datasets (or “projects”). The names of the data collections in the selected dataset will then be loaded into another dropdown box. As the user selects a collection, the run numbers of runs included in the selected collection will be presented in a new dropdown box. At this point, the user can either select some run(s) or make a new collection->run(s) selection. By clicking on "Add selection", the selected collection name (and run number(s)) will be added into an editable text area. Once the user has finished previewing and pre-selecting, he can proceed to the next stop--"Continue to even selection" (See Figure 1).

3.2. ELSSI layer
After the dataset name, collection name(s) and run number(s) have been successfully passed to the ELSSI creation layer, the creation layer builds the ELSSI main page according to the received requests from the parent page. At this point, the user will be presented with a page just like one of the cloned ELSSI’s except the page was built on the fly and the preselected collection name(s) and run number(s) are already entered into the right places (See Figure 2).

![Figure 2](image-url)

Figure 2. The ELSSI layer performs all functions to serve event selection purpose--from providing TAG content for setting up selection criteria, building and executing queries to making service calls to Extract and Skim.
4. Development

4.1. Code Design

ELSSI functions include providing information of available TAGs data sets, guiding users to form valid TAGs database queries, retrieving and presenting user query results in web format or ROOT file format, and invoking other TAG related services. The breakdowns of these functions are in four categories:

1. Database access: PHP classes have been developed to handle connection management, trigger decoding, query forming, validation and execution, TAG contents display, xml inputs assemble, data communication with external services and rendering results;

2. Data rendering: As the underlying databases evolve and the data distribution topology gets extensive, presenting TAG (and related) data for selection has become far more complicated than browsing database tables. A large number of JAVASCRIPT functions have been developed to store collection, run and trigger information in arrays for session use, synchronizing page displays across tabs/pages, dynamically refreshing trigger menu and trigger decoding information as collection and run selection changes, and minimizing the number of database trips.

3. Data processing: The data presentations, validations, query buildings, data importing, data communication with external services are carried out by many AJAX functions. AJAX dynamically present TASK TAG data at the parent layer, set host site for selected collections and runs and provide TAG collections/runs for selection; At the ELSSI layer, AJAX is used to dynamically update physics attributes based on user’s collection selection.

4. Performance/Scalability: Given the limit of memory and array size, optimization has been performed at query building, data retrieval and data storage stages. The resulted iELSSI can provide a full range of event selection service to all past, current and future TAGs datasets with the hosting topology to be heterogeneous to the run level.

4.2. Implementation

Following the site transparent requirement and design strategy, we chose to build one query per (collection_name, site) pair and then eliminate the repeated events across site at the (collection, run) level. In this implementation, the queries were built and executed per (collection_name, site) pair. Query results were saved in associative arrays with collection and site as the array keys. Before presenting the final results to the user, redundant event elimination had to be performed. With this strategy, the Extract/Skim job specification can be sent per collection basis and Extract/Skim can parse and schedule the jobs and then merge the results if necessary.

The above strategy had worked as expected, but there was still room to improve in the aspect of repeated run elimination. Instead of querying all events from each site and then eliminating redundant events at the end, we could use already known query results to avoid repeatedly querying runs from multiple sites at the query building stage. More importantly, the duplicated runs/events are eliminated at the stage of query building. Not only this strategy simplifies the step for removing duplicated events, but also minimizes the data bandwidth usage by avoiding the retrieval of potential duplicates from databases at the first place. After the deliberate query building and execution, the merging of results for each collection across all sites has become relatively easy, especially with all the query data saved in associative arrays.

At one point, the iELSSI event selection page suddenly went blank without any error messages. The cause was found to be the size of the newly uploaded data into the COMA tables iELSSI used. iELSSI retrieves the trigger related data into PHP arrays and then transferred to JAVASCRIPT arrays for trigger menu display and trigger decoding lookup purposes. However, as the COMA tables grew substantially in size, the retrieved data became too large for the PHP memory and subsequently for the JAVASCRIPT arrays. To solve the data volume problem, measures have been taken to restrict the query to only retrieve trigger info for the run range covered by the selected TAG “project” and to use multiple associative JAVASCRIPT arrays for the trigger menu highlighting.
5. Deliverables
iELSSI has been installed at three sites:
1. CERN (Geneva, Switzerland)
2. BNL (Brookhaven, NY, USA)
3. TRIUMF (Vancouver, Canada)

6. Conclusion
The integrated iELSSI was born to save the developers from managing and updating multiple clones at the same time. Now, we only have one copy of iELSSI files to check out/in SVN, only one nightly, only one tag of production code to send to installation sites (CERN, BNL, TRIUMF, etc.). Whenever there is a bug fix or a new feature, it will be updated in only one place. Even if the underlying databases are changed either schema-wise or distribution topology-wise, the new developments in iELSSI will replace the old version and still covers all TAG datasets. As a result, there is no waiting period for a separate iELSSI to be tamed for a new TAG dataset. The newly uploaded TAG data shows up as soon as the uploading is complete, automatically!

If we look at the iELSSI case from the user's perspective, the effect is even more appreciable. We would have to build a dictionary for the user to just find which ELSSI copy to use for her data. The user only needs to know which runs within which data “project” to start his event selection with iELSSI.

7. Acknowledgement
This work is supported in part by the U.S. Department of Energy, Division of High Energy Physics, under Contract DE-AC02-06CH11357.

iELSSI is one of the products from the TAG project. Underneath it is a large array of expertise support provided by the TAG Development Group: Ryan Buckingham, Jack Cranshaw, Tom Doherty, Tulay Donszelmann, Wolfgang Ehrenfeld, Elizabeth Gallas, Julius Hrivnac, Mike Kenyon, David Malon, Marcin Nowak, Mark Slater, Florbela Viegas and Elisabeth Vinek

References
[1] Cranshaw J, et al 2008 Building a scalable event-level metadata service for ATLAS J. Phys.: Conf. Ser. 119 072012, doi:10.1088/1742-6596/119/7/072012
[2] The iELSSI interface URL https://atlas-tagservices.cern.ch/tagservices/TAGBrowser/index.htm.
[3] Cranshaw J, et al 2010 Event selection services in ATLAS J. Phys.: Conf. Ser. 219 042007, doi:10.1088/1742-6596/219/4/042007
[4] Vinek E, et al 2011 Composing Distributed Services for Selection and Retrieval of Event Data in the ATLAS Experiment J. Phys.: Conf. Ser. this conference
[5] Gallas E, et al 2010 An integrated overview of metadata in ATLAS J. Phys.: Conf. Ser. 219 042009, doi:10.1088/1742-6596/219/4/042009
[6] Viegas F, et al 2010 The ATLAS TAGS database distribution and management – Operational challenges of a multi-terabyte distributed database J. Phys.: Conf. Ser. 219 72058, doi:10.1088/1742-6596/219/7/072058
[7] Hrivnac J, et al 2011 ATLAS Tags Web Service calls Athena via Athenaeum Framework J. Phys.: Conf. Ser. this conference
[8] Buckingham R M, et al 2011 Metadata aided run selection at ATLAS J. Phys.: Conf. Ser. this conference