MONITORING OF THE NATIONAL IGNITION FACILITY INTEGRATED COMPUTER CONTROL SYSTEM

J. Fisher, M. Arrowsmith, E. Stout
Lawrence Livermore National Laboratory, Livermore, CA, U.S.A

Abstract

The Integrated Computer Control System (ICCS), used by the National Ignition Facility (NIF) provides comprehensive status and control capabilities for operating approximately 66,000 devices through 2,600 processes located on 1,800 servers, front end processors and embedded controllers. Understanding the behaviors of complex, large scale, operational control software, and improving system reliability and availability, is a critical maintenance activity. In this paper we describe the ICCS message logging framework, with tunable detail levels and automatic rollovers, and its use in analyzing system behavior. ICCS recently added Splunk as a tool for improved archiving and analysis of these log files (about 50GB, or 35 million logs, per day). Splunk now continuously captures all ICCS log files for both real-time examination and exploration of trends. Its powerful search query language and user interface provides interactive exploration of log data to visualize specific indicators of system performance, assists in problems analysis, and provides instantaneous notification of specific system behaviors

ICCS Message Log Framework

A core framework of ICCS is the MsgLog ("message log") API for recording status messages to a file system. Implemented in both Ada and Java, all ICCS software layers use this logging engine.

- Log files are rolling, with old messages on the file system removed at regular intervals
- To allow viewing of start-up behavior, the first log file generated is not part of the rotation process
- A default behavior of four log file, 10,000 lines each, is used if no overriding configuration data is provided
- ICCS typical has 8,500 log files (18gb) after all log rotations

Ada Logging Example

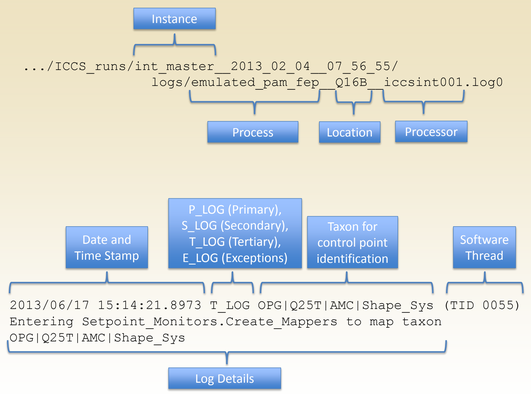
```
Msg_Log_Api.Log_Exception("Unexpected exception
releasing AMC LCU", Get_Taxon (Self));
```

Java Logging Example

```
log.logException("Unable to initialize
controller", ex);
```

Log File Structure

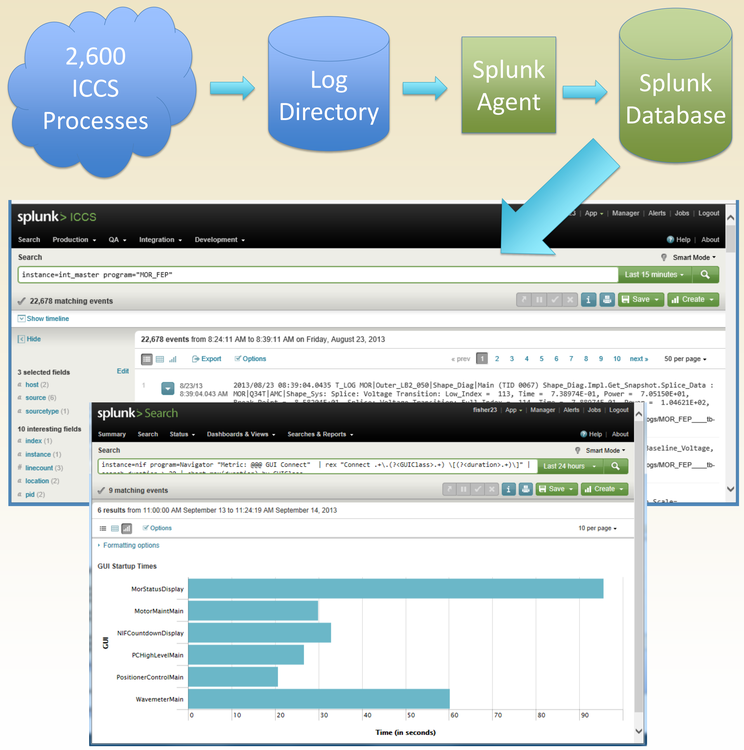
All ICCS log files share a consistent storage location, filename structure, and log format



Capturing ICCS Logs in Splunk

Splunk, developed by a company of the same name, enables ICCS developers to better understand message log data. Splunk is a comprehensive tools for archiving, indexing/searching, parsing, analyzing, and visualizing tremendously large quantities of log file data.

- All log data generated by NIF ICCS, as well as the ICCS QA and Integration environments, is captured and stored in Splunk – about 50GB per day in total
- Typically, a log entry is available in Splunk several seconds after it is stored in the file system. Log retention in the database is currently 45 days.
- ICCS configured Splunk to automatically extract data and time stamp, log level, taxon, thread, instance, process, NIF location, and processor.



Example: Characterizing Log Files

With 2,600 processes which are generating a significant number of errors? How often?

Splunk's queries, using a specialized syntax entered into the above "Search" field, produces raw log data, tables, or charts

These queries work by piping data (with "|") from one command to another. The query below produces the chart on the right:

	program	count	percent
1	MOR_FEP	195	41.666667
2	tapfep	88	18.803419
3	acfep	86	18.376068
4	Config_Services	57	12.179487
5	arcfep	10	2.136752
6	PEPC_FEP	9	1.923077
7	Navigator	8	1.709402
8	Power_Cond_FEP	6	1.282051
9	Segment_Manager	4	0.854701
10	Target2_Diag_FEP	3	0.641026

```
instance="nif" E_LOG | top program
```

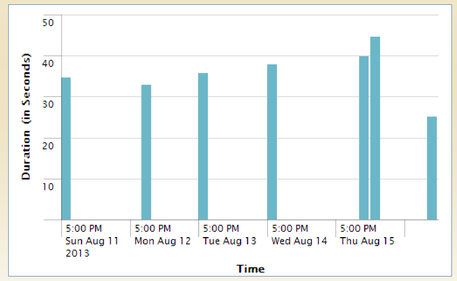
The resulting chart is interactive – clicking on the first row will automatically drill down raw log results, and the Search field will contain:

```
instance="nif" E_LOG program="MOR_FEP"
```

Example: Automatic Alignment Loop Times

For each NIF experiment, ICCS executes a complex series of steps to align all 192 laser beams. Each step, or "loop" takes time to perform. NIF operators have observed that loop times were increasing over several days. Here is an example log entry for the AA_BEAM_TO_TAS_COARSE loop:

```
2013/08/17 02:15:06.959 S_LOG
Segment_Manager|Bu45||@iccsmgrprodbu45 [Interpreter
AA|B453|FOA|SEG-MGR] Metric: [Interpreter AA|B453|FOA|SEG-
MGR] Metric: @@@ SEGMGR AA_BEAM_TO_TAS_COARSE
LOOP_EXECUTION [4.4681] AA|B453|FOA|SEG-MGR
```



The number in brackets is the time in seconds for the loop to complete. The chart above depicts these duration. A restart of several Alignment Front End Processors (FEPs) fixed the issue (the last bar). Here is a Splunk query to chart loop performance, using a regular expression:

```
instance=nif program=segment_manager "AA_BEAM_TO_TAS_COARSE LOOP_EXECUTION" |
rex "LOOP_EXECUTION \[(?<duration>.+)\]" |
timechart span=4h max(duration)
```

Example: Experiment Archiving

Following the execution of an experiment (or "shot"), ICCS device archives experiment results to a database repository. Here is an example log:

```
2013/08/06 23:59:57.701 S_LOG Archive_Server|||@iccsprod0002
[WorkerThread 26 (Database Transaction Mgr)] Metric:
[WorkerThread 26 (Database Transaction Mgr)] Metric: @@@
request_Data_Archive Worker [0.0297] shotId=C130805-AA
diagId=AggregatePeriodicArchiver requestId=137585879767000.000
formatter=nif.subsystems.cts.db.SIM960AggregateFormatter done
```

	formatter	avg(duration)
1	nif.subsystems.pcs.db.CurrentMonFormatter	47.554560
2	nif.common.energy_diag.db.CalDataFormatter	0.300675
3	nif.subsystems.pcs.db.ChargeProfilesFormatter	0.148952
4	nif.subsystems.pcs.db.ShotDataFormatter	0.107205
5	nif.subsystems.pcs.db.ShotSetupFormatter	0.081474

Focusing on a specific NIF experiment, where the Shot ID is N130804-001-999, the table above depicts average archiving performance across all NIF locations (e.g. all beamlines). This information is valuable in improving the overall post-shot data archiving time.

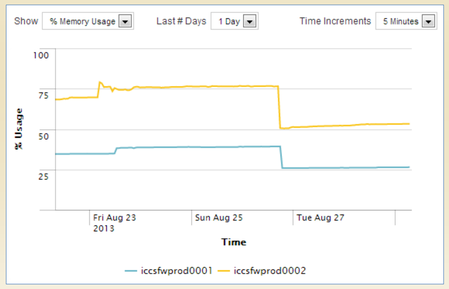
Splunk automatically recognizes the pattern *name=value*, so the shotID field is automatically extracted. The duration field requires a regular expression. The query to generate this table is below.

```
instance=nif program=Archive_Server "@@@ request_Data_Archive" |
rex "Worker \[(?<duration>.*)\]" |
search shotId=N130804-001-999 |
stats avg(duration) by formatter
```

Example: Server Monitoring

All ICCS Framework and Supervisory servers are monitored by Oracle Enterprise Manager (OEM). Splunk is used to analyze OEM data in Oracle database tables just like log data. The Splunk query below gathers OEM metrics data and plots the results in a timeline (shown at right) for two Frameworks servers.

This query resides in an XML file, used by Splunk for creating complex dashboards. The query is parameterized (`$selectedDays$`, `$selectedReports$`, `$increment$`), based on user fields on the dashboards.



The two hosts in this graph are both Oracle Virtual Machines (OVMs); as a result of this analysis, memory for both were increased (as can be seen in the graph). OVM technology allowed this to be done without a server restart.

```
| dbquery OEM "SELECT regexp_replace(d.target_name, '\\..*', '') AS Host, d.collection_timestamp, d.metric_column as
Metric_Type, TO_NUMBER(d.value) as Metric FROM sysman.mgmt$metric_details  d WHERE d.collection_timestamp >= SYSDATE -
$selectedDays$ and d.target_type = 'host' and d.target_name like 'iccsfwprod%' and d.metric_column = '$selectedReport$'" |
rename COLLECTION_TIMESTAMP to _time |
timechart span=$increment$ max(METRIC) by HOST
```

LLNL-POST-644196

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.