Dashboard: A Tool for Integration, Validation, and Visualization of Distributed NLP Systems on Heterogeneous Platforms

Pawan Kumar
Expert Software Consultants Ltd
New Delhi, India
hawahawai@gmail.com

Rashid Ahmad
LTRC, IIIT
Hyderabad, India
rashid.ahmed@research.iiit.ac.in

B. D. Chaudhary
CSED, MNNIT
Allahabad, India
bdc@mnnit.ac.in

Mukul K. Sinha
Expert Software Consultants Ltd
New Delhi, India
mukulksinha@gmail.com

Abstract

Dashboard is a tool for integration, validation, and visualization of Natural Language Processing (NLP) systems. It provides infrastructural facilities using which individual NLP modules may be evaluated and refined, and multiple NLP modules may be combined to build a large end-user NLP system. It helps system integration team to integrate and validate NLP systems. The tool provides a visualization interface that helps developers to profile (time and memory) for each module. It helps researchers to evaluate and compare their module with the earlier versions of same module. The tool promotes reuse of existing modules to build new NLP systems. Dashboard supports execution of modules that are distributed on heterogeneous platforms. It provides a powerful notation to specify runtime properties of NLP modules. It provides an easy-to-use graphical interface that is developed using Eclipse RCP. Users can choose an I/O perspective (view) that allows him better visualization of intermediate outputs. Additionally, Eclipse RCP provides plugin architecture; hence customized end-user specific functionality can be easily added to the tool.

1 Introduction

Dashboard is a tool for integration, validation, and visualization of Natural Language Processing (NLP) systems on heterogeneous platforms. Dashboard tool has been designed to build NLP systems reusing multiple heterogeneous (or homogeneous) NLP components.

Dashboard provides a common representation called *Shakti Standard Format* (SSF) (Akshar Bharati et al. 2007), a tree data structure, for storing linguistic information (analysis) produced by an NLP module in *attribute/value* pairs called *feature structure*. The SSF format has both *in-memory* representation as well as *stream* representation. They are inter-convertible using a *Reader* (stream to memory) and *Printer* (memory to stream). The in-memory representation is good in speed of processing, while the stream representation is good for *portability*, *heterogeneous platforms*, and *flexibility*, in general.

The tool provides SSF API for integrating NLP modules; modules use APIs for accessing in-memory data in multiple programming languages. For integration of existing modules SSF adapters are used. These adapters transform varying input/output representations to SSF format.

Dashboard has a frontend called *Dashboard* and a backend called *Dashboard Runtime*. Dashboard provides a graphical interface for setting up and configuring NLP systems. Dashboard Runtime performs the tasks of *coordination* and *communication* between the NLP modules.

A. Dashboard – the frontend

The Dashboard frontend provides a mechanism to setup and configure the NLP Systems that are based on Blackboard Architecture (Hayes-Roth 1985, Sangal 2004) runs on the Dashboard platform.

B. Dashboard Runtime

Dashboard Runtime performs the tasks of coordination and communications between the modules.

C. Dashboard as a Visualization Tool

Dashboard provides visualization and debugging functionality to researchers and developers.

This tool is now released and is being used by consortium members at 11 research institutions. Eighteen machine translation systems which translate between 9 pairs of Indian languages have been integrated and tested using this tool

The Companion Volume of the Proceedings of IJCNLP 2013: System Demonstrations, pages 9–12,
Nagoya, Japan, 14-18 October 2013.

(Sampark MT 2013, Anthem 2010). A comprehensive description of Dashboard tool is available in (Pawan et al. 2010, 2013).

2 Dashboard Features

Distinguishing features of Dashboard tool are:

A) NLP System can be partitioned, and different partitions (comprised of one or more modules) can reside on heterogeneous platforms on distributed machines.

B) Dashboard Tool provides powerful notation for describing runtime properties of individual modules as well as complete NLP system.
   - New modules can be added/removed either using user-interface or using declarative notation in specs file,
   - Switch runtime modes (*speed*, *debug* or *stepwise*) by the user/developer
   - *Conditional-run* for each module, i.e., at runtime user can skip execution of a module depending on (presence or absence of a *feature structure* in SSF data) input data,
   - *Triggers* for modules to promote robustness, e.g., set timeout for module that goes into an infinite loop.

C) Visualization interface is developed using Eclipse Rich Client Platform (RCP). RCP provides plugin architecture (McAffer et al. 2010). Developers can build user-defined plugins leveraging this plugin architecture.

D) Unlike other frameworks like GATE, UIMA and OpenNLP, Dashboard linguistic pipeline is created by specifying module properties in attribute/value pairs, which is compiled to generate an efficient runtime.

E) Similar to GATE, UIMA and OpenNLP, Dashboard can output linguistic analysis in SSFXML format that can be transformed to various formats using XSLT.

3 Sampark MT System: An Application Using Dashboard

We provide here screen shots of Punjabi-Hindi Sampark MT System (*sampark-pan2hin*). Figure 1 shows the normal view of the Dashboard. For reasons of clarity, we have labeled the icons in Figure 1, with numbers in red. On the Tool bar (labeled-2) we have *Run* button (labeled-1), and *View* button (labeled-6) on extreme right.

Below the Tool bar on extreme left is the *Application Explorer* pane (labeled-8). It shows the list of Dashboard applications (*sampark-pan2hin* system and *sampark-hin2pan* system). Just below that, i.e., *sampark-pan2hin* system all the modules of *sampark-pan2hin* system is listed. On the right of this pane we have a tabbed window where input text (labeled-3), in Punjabi language, *punjabi.txt* (visible), and output text (labeled-4), in Hindi language, *punjabi-output.txt* (this tab is invisible) are shown. Below this pane input/output from a module, *lexical-transfer* is shown (labeled-9 and labeled-10). Label-11 shows the error log if any from the system. Label-12 shows the selected module name along with selected sentence number. Label-13 shows the time profile for the selected module. Label -15 and 16 show tool bar for switching views of input and output data useful to computational linguists. The user can select views from a list of views i.e., SSF or XML or Native.

Figure 2, shows complete translation of Punjabi text into Hindi. Left pane shows the input text composed of 10 sentences, written in source language Punjabi, and the right pane shows translated output text, as the output from Sampark system written in target language Hindi.

Once the system is executed completely, the session can be saved for future analysis. In case the user wants to analyze the intermediate output of any specific module for a specific sentence, he can do so through Dashboard. For this he has to first expand the module list in the Application Explorer pane. Select the module whose intermediate output he wants to analyze. Once the user selects the module, he gets a Dashboard view as shown in Figure 3.

Figure 3, shows input to lexical-transfer module in the left pane, and output from lexical-transfer module into the right pane for sentence number 5 in SSF format. Time taken to execution of lexical-transfer module is shown at the bottom of the pane earlier labeled-13 in Figure 1.

To start any application/project system has to be configured/setup, which is done chronologically earlier but is being explained now. Figure 4 shows how a Sampark system is setup and configured for a new application/project. In the Application Explorer pane user has to add an application first. Once he adds an application, *DashboardSpec.xml* icon appears in the Application Explorer pane. When the user selects *DashboardSpecs.xml* icon, a tab window, earlier la-

beled-5 becomes visible, 'Applications Module Specification' pane is opened. On the bottom of this pane there are 5 tabs, viz., *Overview*, *Global Prop.*, *Runtime*, *Modules*, and *DashboardSpec.xml*. *Overview* tabs gives an overview about system configuration and setup. '*Global Prop*' tab configures the global properties for Dashboard Runtime, like, location of SSF API for various languages. Currently SSF API is available for C/C++, Perl, Python, and Java. *Runtime* tab allows for adding runtime arguments to the Dashboard (like UNIX command line parameters). *Modules* tab allows defining runtime properties of each module. Figure 4 shows the runtime properties of Punjabi Morph Analyzer. In the figure, only standard properties are defined. In the *DashboardSpec.xml* tab, user can manually edit the *DashboardSpec.xml* file.

Acknowledgments

We sincerely thank TDIL group, Dept. of IT, Govt. of India in granting and supporting such a challenging project. All NLP researchers and software engineers of the participating institutions (viz., IIIT Hyderabad, IIT Mumbai, IIT Kharagpur, Central University Hyderabad, AU-KBC Research Center Chennai, C-DAC NOIDA, Jadavpur University, Kolkata, IIIT Allahabad, and IISc Bangalore) need special thanks who have been using this tool and have helped in identifying new requirements for its improvement. Their continuous feedback has not only improved its functionality, but also stabilized it as a product.

We thank our colleagues, Rambabu, Phani, Avinash, and Sanket without whose tireless effort current version of the tool would not be possible.

We sincerely thank Prof. Rajeev Sangal, who allowed us to work on such an innovative project, because 'Dashboard as NLP Tool' was primarily his idea.

References

Akshar Bharati, Rajeev Sangal and Dipti M. Sharma. 2007. *SSF: Shakti Standard Format Guide*, LTRC, IIIT, Hyderabad, Report No: TR-LTRC-33.

Sampark MT. *Machine Translation System among Indian languages*. http://sampark.org.in, last accessed on 15-Feb-2013.

G. Anthes. 2010. *Automated Translation of Indian Languages*. CACM, vol. 53 (1), pp. 24-26.

B. Hayes-Roth. 1985. *Blackboard Architecture for Control*, Art. Intelligence.

R. Sangal. 2004. *Architecture of Shakti Machine Translation System*. IIIT Hyderabad.

Pawan Kumar, et al. 2010. *Dashboard: An Integration and Testing Platform based on Blackboard Architecture for NLP Applications*. Proc. of IEEE 6th Intl. Conf. on Natural Language Processing and Knowledge Engineering, Beijing, China. Report No: IIIT/TR/2010/84.

Jeff McAffer. 2010. *Eclipse Rich Client Platform*, Second Edition, Addison Wesley.

Pawan Kumar, et al. 2013. *Enriched Dashboard: An Integration and Visualization Tool for Distributed NLP Systems on Heterogeneous Platforms*. Proc. of IEEE 13th Intl. Conf. on Computational Science and Applications, Ho Chi Minh City, Vietnam.

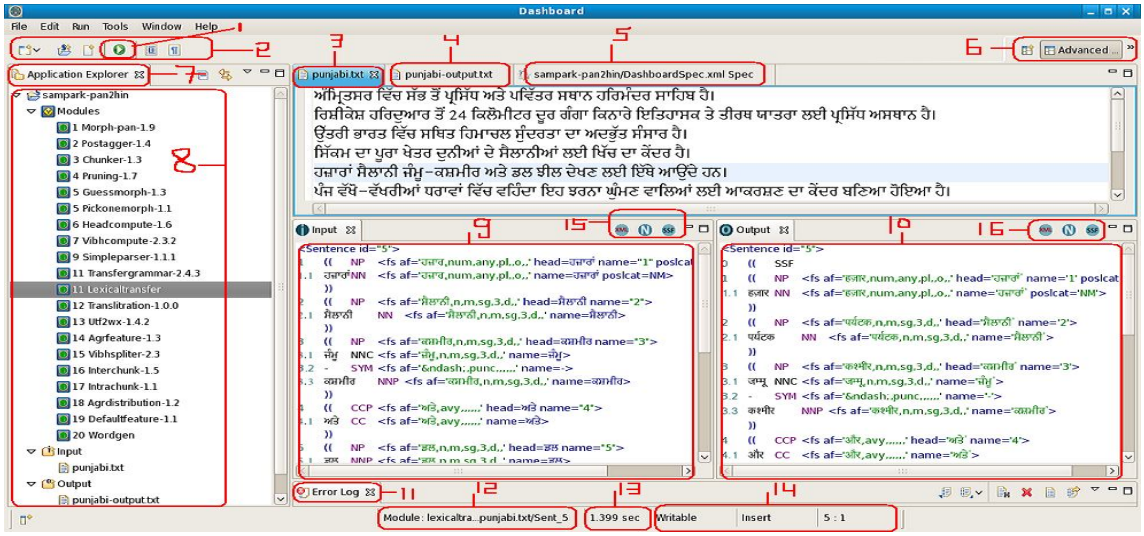

Figure 1: Normal view of Dashboard Tool, icons/panes/buttons are labeled to describe the tool's functionality.

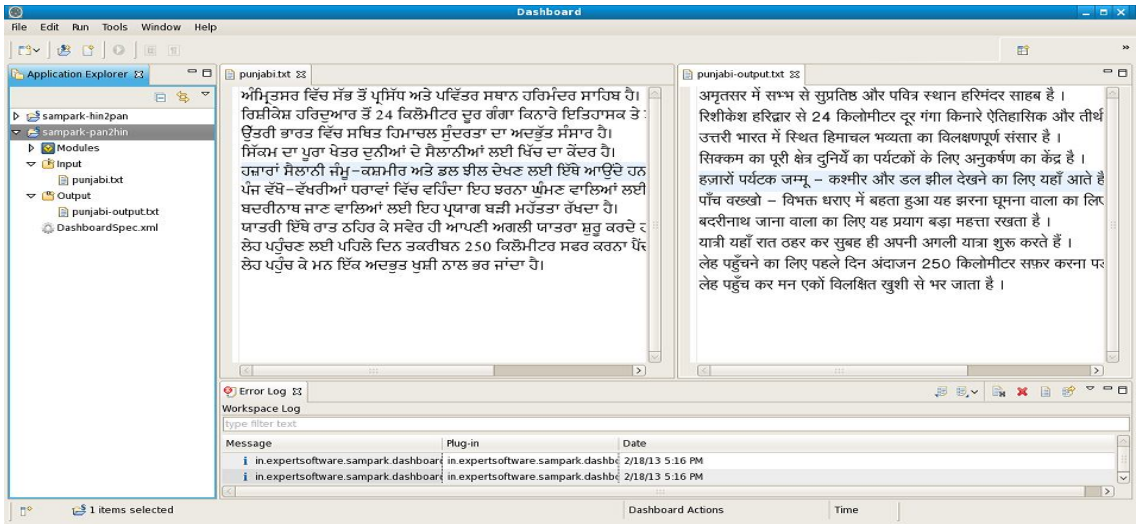

Figure 2: Shows Punjab-Hindi MT system, shows input and output text for 10 sentences.

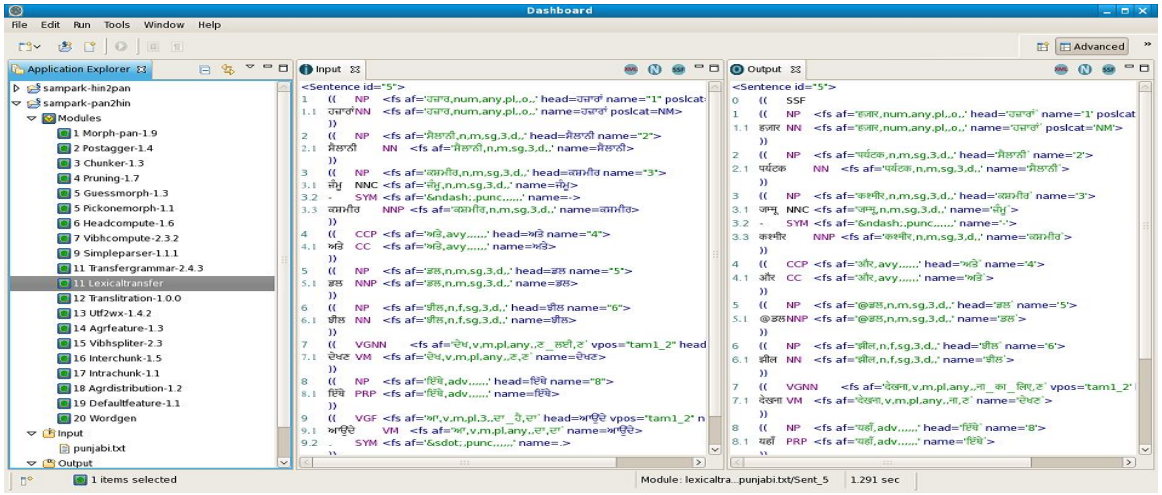

Figure 3: Module level input and output for *lexical-transfer* module, data in SSF format for sentence number 5.

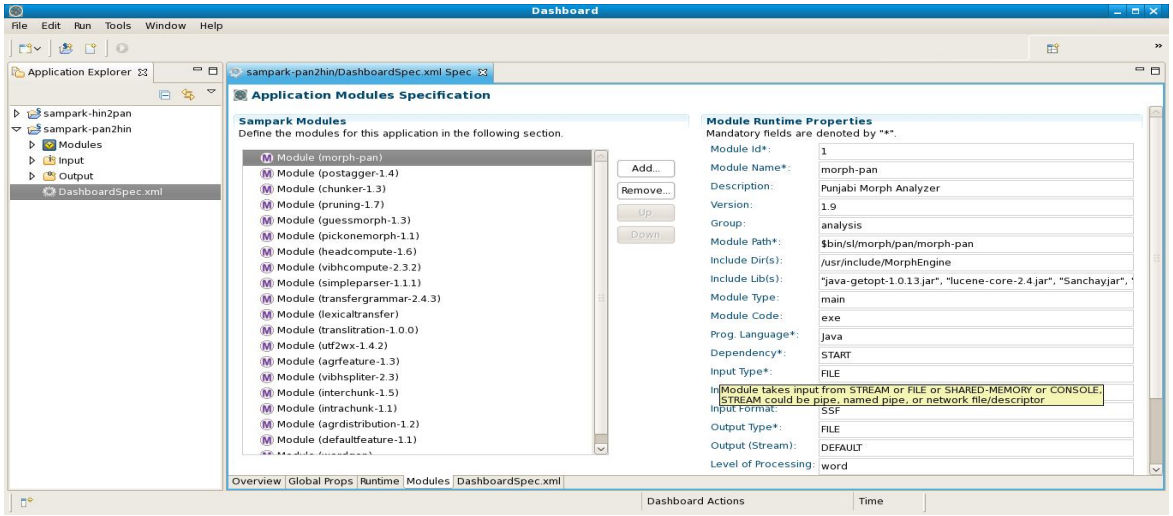

Figure 4: Application Setup and Configuration view. Right pane shows runtime properties for *Punjabi Morph Analyzer*. Also shows help text, if the mouse is hovered on the text labels.