

LAUNCHPADS: A System for Processing Ad Hoc Data

Mark Daly
Princeton University
mdaly@Princeton.EDU

Mary Fernández
Kathleen Fisher
AT&T Labs Research
mff,kfisher@research.att.com

Yitzhak Mandelbaum
David Walker
Princeton University
yitzhakm,dpw@CS.Princeton.EDU

An Introduction to PADS. Ideally, any data we ever encounter will be presented to us in standardized formats, such as XML. Why? Because for formats like XML, there are a whole host of software libraries, query engines, visualization tools and even programming languages specially designed to help users process their data. However, we do not live in an ideal world, and in reality, vast amounts of data is produced and communicated in *ad hoc formats*, those formats for which no data processing tools are readily available. Figure 1 presents a small selection of ad hoc data sources. As one can see, ad hoc data exists in a very wide variety of fields and the users range from network administrators to computational biologists and genomics researchers to physicists, financial analysts and everyday programmers.

Programmers often deal with this data by whipping up one-time Perl scripts or C programs to parse and analyze their data. Unfortunately, this strategy is slow and tedious, and often produces code that is difficult to understand, lacks adequate error checking, and is brittle to format change over time. To expedite and improve this process, we developed the PADS data description language and system [2, 3]. Using the PADS language, one may write a declarative description of the structure of almost any ad hoc data source. The descriptions take the form of types, drawn from a dependent type theory. For instance, PADS base types describe simple objects including strings, integers, floating-point numbers, dates, times, and ip addresses. Records and arrays specify sequences of elements in a data source, and unions, switched unions and enums specify alternatives. Any of these structured types may be parameterized and users may write arbitrary semantic constraints over their data as well.

Once a programmer has written a description in the PADS language, the PADS compiler can generate a collection of format-specific libraries in C, including a parser, printer, and verifier. In addition, the compiler can compose these libraries with generic templates to create value-added tools such as an ad hoc-to-XML format conversion tool, a histogram generator, and a statistical analysis and error summary tool. Finally, PADS has been composed with the GALAX query engine [6, 4, 5] for XQuery to create PADX [1], a new system that allows users to query and transform any ad hoc data source as if it was XML, without incurring the performance penalty that usually results when one converts ad hoc data into a much more verbose XML representation.

While the PADS language provides an extremely versatile means of creating tools for processing ad hoc data, it is nevertheless a *new* language and learning a new language is time-consuming for anyone, especially for computational biologists or other scientists for whom programming is not their primary area of expertise. To ease the way for novice PADS users, we developed LAUNCHPADS, a new tool that provides access to the PADS system without requiring foreknowledge of the PADS language itself. Hence, LAUNCHPADS graphic interface will also help more experienced PADS users to shorten their development cycle and provides a con-

Name : Use	Representation
Web server logs (CLF): Measure web workloads	Fixed-column ASCII records
CoMon data: Monitor PlanetLab Machines	ASCII records
Call detail: Fraud detection	Fixed-width binary records
AT&T billing data: Monitor billing process	Various Cobol data formats
Netflow: Monitor network performance	Data-dependent number of fixed-width binary records
Newick: Immune system response simulation	Fixed-width ASCII records in tree-shaped hierarchy
Gene Ontology: Gene-gene correlations	Variable-width ASCII records in DAG-shaped hierarchy
CPT codes: Medical diagnoses	Floating point numbers

Figure 1. Selected ad hoc data sources.

venient way for experts to quickly create any of the data processing tools they need.

LaunchPads. LAUNCHPADS combines mechanisms for graphically defining structure and semantic properties of ad hoc data, for translation of this definition into PADS code, and for compilation/execution of the generic tools that operate over ad hoc data. More specifically, LAUNCHPADS breaks definition of an ad hoc data format and generation of data processing tools into the following steps. Figure 2 presents a screenshot of LAUNCHPADS being used to construct a data description for a web-server log format.

- 1. Selection of sample data from which to build the description.** Creation of a definition within LAUNCHPADS begins when a user loads sample data into the graphical interface. In Figure 2, web log data (beginning with the IP address 207.136.97.49 . . .) appears in the top right hand corner of the picture. A user then selects a row of data to work on in the LAUNCHPADS *gridview* immediately below.
- 2. Iterative refinement in the gridview.** Once in the gridview, users may specify descriptions for regions of text using a highlighting scheme. The color assigned to a region represents the description class (base or composite) and region boundaries. Structure within a definition is represented through a series of refinement steps: composite regions are broken down and level after level, thereby allowing for nested elements (Figure 2 shows four nesting levels). The refinement process bottoms out when one reaches an atomic description such as a character string, IP address or date. Once all regions have been given a base type in the gridview, LAUNCHPADS will generate a *tree-view* of the definition for further processing.

