

PuppyIR: Designing an Open Source Framework for Interactive Information Services for Children

Leif Azzopardi, Richard Glassey,
Mounia Lalmas, Tamara Polajnar
Dept. Computing Science
University of Glasgow
Scotland, U.K.

{leif, rjg, tamara, mounia}@dcs.gla.ac.uk

Ian Ruthven
Dept. of Computer and Information Sciences
University of Strathclyde
Scotland, U.K.
ir@cis.strath.ac.uk

ABSTRACT

One of the main aims of the PuppyIR project is to provide an open source framework for the development of Interactive Information Retrieval Services. The main focus of the project is directed towards developing such services for children, which introduces a number of novel and challenging issues to address (such as language development, security, moderation, etc).

In this poster paper, we outline the preliminary high-level design of the open source framework. The framework uses a layered architecture to minimize dependencies between the user-side concerns of interaction and presentation, and the system-side concerns of aggregating content from multiple sources and processing information appropriately. Each layer will consist of a series of interchangeable components, which can be interconnected to form a complete service. To facilitate the construction of diverse information services, a dataflow language is proposed to enable the assembly of the components in an intuitive and visual manner. One of the design goals of the architecture, and ultimate measures of success, is to provide a “lego” style building block environment in which researchers and developers of any age can build their own information service.

This poster provides the starting point for the design of the framework and aims to seek comments, feedback and suggestions from the community in order to improve and refine the architecture.

1. INTRODUCTION

Today children are exploring the Internet from a very early age. However, much of the content and services available on the Internet have been designed for adults. And so while many children actually possess better computing skills than their parents, they are also left vulnerable to the darker side of the net (in the worst case) or overwhelmed and overloaded when searching for information available online [7]. Furthermore, they also have to interact and engage with

systems and tools that are not specifically designed to consider or develop their cognitive, emotional and intellectual abilities [3].

Consequently, many children experience difficulties during information seeking, as they have to contend with a number of challenges: non-intuitive search interfaces; unmoderated and unverified content; language and understanding issues; and structured and prescribed interaction styles. In order to help children effectively, efficiently and enjoyably engage with information, entertainment, friends, and so forth online, it is important to provide access to such information services in ways that are consistent with their learning, cognitive development and curriculum [3]. To support the investigation of such challenges, the PuppyIR project aims to provide the infrastructure to construct and build interactive information services using an open source framework.

The remainder of the poster is as follows: In the next section we discuss the aims of the PuppyIR project, then we outline the high-level design of the open source framework, before providing an example of constructing an information service. Finally, we conclude with a brief summary.

2. PUPPYIR PROJECT

PuppyIR¹ is an European Union project that is funded under the European Community’s Seventh Framework Programme. The project will run over the next three years and involves seven partners from within European Union. The aims of the project are as follows:

1. Investigation and promotion of new interaction paradigms for children’s access systems.
2. A number of interrelated research questions will be addressed that contribute to the interaction of children with information services. The focus will be on presentation, mining and language issues for the appropriate development of these information services.
3. An open source framework will be created to support the development of common interchangeable components to support the realization of information services for children.
4. Identification of appropriate evaluation measures and develops an evaluation framework for information services for children.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

HCIR '09 Washington, USA

Copyright 200X ACM X-XXXXX-XX-X/XX/XX ...\$10.00.

¹<http://www.puppyir.eu/>

In this poster paper, we shall focus on the 3rd aim of this ambitious and exciting project and outline the preliminary design for the high level architecture of the PuppyIR Open Source Framework, as a way to open up the project to the wider community and seek input and feedback on the design and architecture.

3. OPEN SOURCE FRAMEWORK DESIGN

One of the main design goals of this project is to create an open source framework that supports the rapid development of child-friendly Interactive Information Retrieval (IIR) services. The provision of such a framework enables both system developers and researchers to design and create different types of IIR systems using a common library of components and configurations, and to do so quickly and cost-effectively.

With that in mind, the PuppyIR Open Source Framework (PIR-OSF) aims to deliver an open, reconfigurable and reusable software framework. It avoids unnecessary duplication of effort and allows for rapid development of multiple independent IIR services. Finally it provides an effective evaluation environment for ongoing HCI and IR research, based around an open and common toolset.

To achieve these aims, the design of PIR-OSF establishes three fundamental abstractions for modeling a generic IIR service:

1. **Layered Architecture:** provides the separation of concern between the common aspects of an IIR service (information sources, information processing, visualization and interaction) that enables multiple environments and application scenarios to be addressed using a single framework.
2. **Rich Information Objects:** a flexible abstraction for any type of information content (text, images, audio, movie, mixed, etc.) and its accompanying meta-data required by an IIR service.
3. **Dataflow Language for Information Processing:** allow both users and developers to reuse, visually reconfigure alternative information processing networks to deliver the most appropriate information.

The following sections discuss these abstractions and their implications upon the design of the PIR-OSF.

3.1 Layered Architecture

In its most basic form, an IIR service allows users to interact by submitting their queries, then presenting them with the static results (e.g. via a search engine) or a stream of information (e.g. a RSS feed) using an appropriate form of presentation. The architecture of the PIR-OSF represents this high-level system view using a series of independent, stacked layers.

This design decision reflects the different requirements of building information services for multiple environments and application scenarios. More fundamentally, it recognizes the close relationship that exists between human-computer interaction and information retrieval. By providing a common framework, researchers from both fields can conduct research within their appropriate layer(s) and reuse components from a pre-existing library built by others. Thus, the entry barrier to building a complete system is dramatically

lowered, reducing the effort for both systems' designers and the HCIR research communities.

A concrete illustration of the importance of a layered architecture is to consider a unified museum guide for children. Results of a query issued at a public static terminal or from their own mobile device could be projected onto a nearby wall-mounted display, taking advantage of the extra screen space and perhaps touch-oriented interaction. Should no display be nearby, the user could still use their mobile device, with the same underlying system, but the presentation of and interaction with the results would be vastly different. To accommodate these differences, a flexible architecture is essential. Figure 1 shows the layered architecture of PIR-OSF. A conceptual divide separates the lower system-side layers from the upper user-side layers.

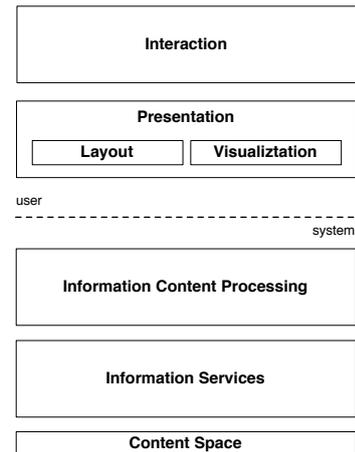


Figure 1: Layered architecture of the PuppyIR Open Source Framework

System-side Layers: At the bottom of the stack, the Content Space layer represents all of the potential information sources that are indexed and made available to the rest of a system. The Information Services layer encompasses the various interfaces to online services, such as Google, Yahoo, Bing, Youtube, Flickr, Twitter (uncooperative services) and so on, as well as content indexed by offline services such as Lemur, Lucene and Terrier (cooperative services). This layer is responsible for using the respective API of each service and specifies the interface for creating result wrappers for each service, such that results can be returned as collections or streams of Rich Information Objects (RIO).

The Information Content Processing layer manipulates the RIOs by passing them through a network of linked components called Information Content Processors (ICP), configured using a dataflow-oriented approach. For example, a typical ICP may moderate the content of a RIO and return the modified result for further processing. More complex ICPs may act as aggregators and distributors of multiple RIOs, as shown in Fig. 2. Alternatively, two summarizer ICPs could be evaluated by swapping them into the same ICP network, without changing any other aspect of the configuration. This networked aspect provides complete flexibility with the configuration, making it suitable to develop a wide range of systems.

User-side Layers: The upper layers relate to the user-

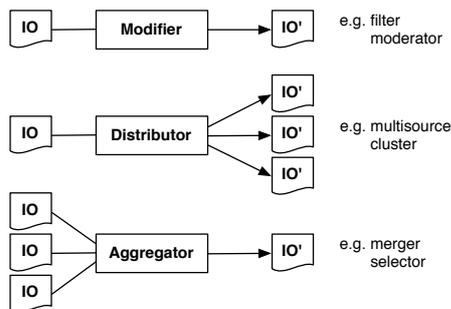


Figure 2: Information Content Processor (ICP) configurations

side concerns; how RIOs are presented, and how the user can interact with the system in terms of query formulation and interactive feedback. The Presentation layer deals with the overall layout of containers of RIOs (e.g. a webpage that has several blocks for separating image and textual results) and the visualization of RIOs within the containers (e.g. a ranked list of results or a carousel of images). The Interaction layer corresponds to the variety of modes and modalities that a user may utilize when using the system, ranging from the familiar textual and graphical interfaces, onwards to more advanced speech, touch and tangible forms of interaction.

3.2 Rich Information Objects

The RIO is an abstraction for all kinds of information that an information service may produce or need to consume. It exposes a public interface that allows the components of PIR-OSF to inspect, manipulate, and exchange it with other components. This abstraction also permits the creation of new content types without needing to modify existing components of the PIR-OSF. For example, new RIOs could be created to mix multiple types of information together as required by a service.

Besides representing information, such as text, images and movies (or combinations), each RIO contains metadata, which either has been extracted from the information (e.g. the title of a web page), or added by a component of PIR-OSF as part of its processing (e.g. a readability rating). The range of metadata is not fixed and is fully extensible depending upon the system needs.

This approach is crucial to providing a common object that all components can inspect, annotate, transform and exchange. Whilst this type of information wrapping inevitably creates overhead costs in terms of time and space when processing objects, it is crucial for the final systems abstraction of PIR-OSF: the dataflow oriented approach to processing RIOs.

3.3 Dataflow Language for Information Content Processing

Each service built using the PIR-OSF will have different information processing needs. Since there are a lot of differences between users and their specific needs, as they are children (i.e. lots of variation in age, language skills and language, knowledge, tastes, moods, etc), then the flexibility to configure the information services is paramount. Example

information content processing units that will be provided to support the information seeking of children will include: moderation components which filter the content ensuring it is suitable and/or appropriate; summarisation components which will re-factor content to suit the child's reading ability, classification components to aggregate and facet information depending on their context. Consequently, the PupyIR project will focus on developing information processing components which are designed for children.

The ability to chain together different ICPs is therefore critical towards meeting the different IIR service requirements. One approach might be to treat the chaining of ICPs as a pipe and filter architecture [6], however this reduces the configuration of ICPs to a single, serial pipeline and does not allow a rich network of ICP components to be assembled. Instead, PIR-OSF adopts a dataflow language oriented-approach.

The essence of dataflow programming languages is that a program can be expressed as a directed graph, with nodes acting as primitive instructions (such as arithmetic and comparative operations) and directed arcs representing the data dependencies between these instructions [5] (see Fig. 3).

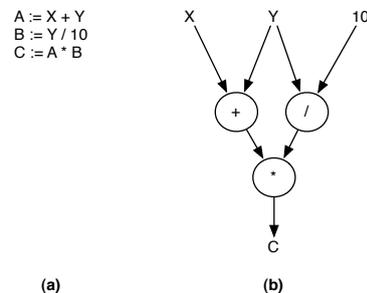


Figure 3: A simple program (a) and its dataflow equivalent (b)

Historically, dataflow languages have been closely linked to visual programming, however the infancy of display and interaction technology hampered their development [4]. With advances in user interfaces, increasingly visual approaches are being adopted for dataflow programming, e.g. the MIT Scratch programming environment², Apple's Quartz Composer³ for graphics programming and Yahoo! Pipes for mashing web data together⁴. We adopt this approach for the PIR-OSF, where ICPs replace the instructions and RIOs replace the data. There are three main justifications for using this design choice:

Visualization: Early in the history of dataflow programming, it was recognized that dataflow languages could be easily adopted by novice users in order to communicate and construct programs [1]. Given the intuitiveness of such languages, we envisage that not only could systems designers easily (re)-configure the system using visual tools, but the users themselves (in this case children) could become the designers of their own information services; allowing them to better fit their own individual information seeking behavior.

Parallelism: Dataflow programming was proposed as a

²<http://scratch.mit.edu>

³<http://developer.apple.com/graphicsimaging/quartz>

⁴<http://pipes.yahoo.com/pipes>

solution to the von Neumann execution model, in which an instruction can only be executed when the program counter reaches it, even if it could be executed earlier. Many scenarios exist where multiple sources of information can be accessed in parallel and concurrently processed without affecting the final results. Given the importance of responsiveness to the user experience [2], parallelism should be exploited.

Verifiability: As the development of any concurrent process tends to be complicated, the use of dataflow programming may help reduce this effort. Networks of ICPs, and the flow of RIOs through them, can be formally modeled, simulated and verified, before spending time and effort trying to develop, debug and maintain a complex concurrent system.

4. DESIGNING AN INTERACTIVE INFORMATION SERVICE WITH PIR-OSF

By way of illustration, we present the high-level design of an IIR service using PIR-OSF. We focus mainly upon the component interaction and the flow of information processing for the system-side aspects of the IIR service. The user-side aspects of the system, presentation and interaction, can be assumed to take the form of a web page with aggregated results (a page with a block reserved for web search results, and another for combined image and video results).

Figure 4 shows the IIR service, with two independent ICP networks configured, and the layered architecture overlaid for reference. On the left-hand side, a simple single-path network has been created. In it, a query is submitted, moderated, then sent to the Google search engine using its API. The results returned are wrapped into a set of RIOs at the Information Services layer, before being passed through the moderation and summarizer ICPs. This returns a modified result set (RIO') that can then be presented to the user.

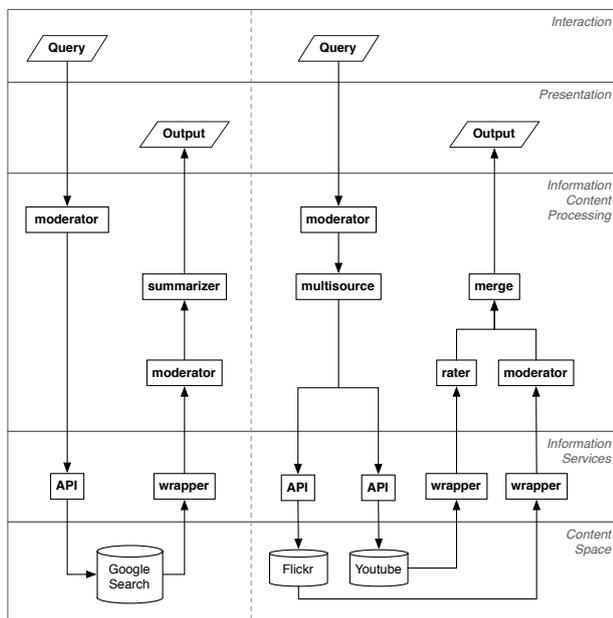


Figure 4: High-level design of a single IIR service with two independent network configurations of ICPs

In contrast, the right-hand side of Fig. 4 shows a more

complex network of ICPs. The query is passed through a moderator ICP and a multi-source ICP, which can be configured to distribute its input to multiple recipients (see distributor ICP in Fig. 2). In this case, it submits the same query to both Flickr and Youtube. This returns two result sets, which are processed in parallel (one moderated for adult content, whilst the other is ordered by ratings), then merged together into a single result set for presentation to the user. As the same query can be injected into both ICP networks, output can be presented together to the user, in the most appropriate form for the IIR service.

In this example trivial combinations of ICPs were used; more complex networks could be created for specific IIR services. By using a dataflow-oriented approach, IIR services can more easily be reconfigured and customised by both users and systems developers. Furthermore, this approach permits the evaluation of similar ICPs (e.g. two summarizers), by swapping them into an active network, whilst monitoring the user experience and response.

5. SUMMARY

In this poster paper, we have outlined the preliminary high-level design of the PuppyIR Open Source Framework for constructing information services. Key to the design is the layered architecture used to assemble the desired service functionality. This is underpinned by a dataflow language that models the flow of Rich Information Objects through a flexible network of Information Content Processors, from the content space to the presentation layer. It is envisaged that this plug and play architecture will enable the rapid development of information services, whilst also being highly extensible and configurable.

Acknowledgements: This research is funded by the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement no. 231507.

6. REFERENCES

- [1] A. L. Davies. Data-driven nets - a class of maximally parallel, output-functional program schemata. Technical report, Burroughs, San Diego, CA, 1974.
- [2] W. O. Galitz. *The essential guide to user interface design: an introduction to GUI design principles and techniques*. John Wiley and Sons, 2002.
- [3] H. E. Jochmann-Mannak, T. W. C. Huibers, and T. J. M. Sanders. Children's information retrieval: beyond examining search strategies and interface. In *The 2nd BCS-IRSG Symposium: Future Directions in Information Access*, 2008.
- [4] W. M. Johnston, J. R. Paul-Hanna, and R. J. Millar. Advances in dataflow programming languages. *ACM Computing Surveys*, 36(1), March 2004.
- [5] P. Kosinski. A data flow language for operating systems programming. In *Proceedings of the ACM SIGPLAN-SIGOPS Interface Meeting*, 1973.
- [6] R. N. Taylor, N. Medvidovic, and E. Dashofy. *Software Architecture: Foundations, Theory, and Practice*. John Wiley and Sons, 2009.
- [7] Y. Zhang. How children find information on the internet: An empirical study and its implications. In *The 3rd Annual Research Conference of the Research Center for Educational Technology (RCET)*, Kent, OH., 2002.