XIML: A Common Representation for Interaction Data

Angel Puerta and Jacob Eisenstein
RedWhale Software
277 Town & Country
Palo Alto, CA USA
+1 650 321 0348
{puerta,jacob}@redwhale.com

ABSTRACT

We introduce XIML (eXtensible Interface Markup Language), a proposed common representation for interaction data. We claim that XIML fulfills the requirements that we have found essential for a language of its type: (1) it supports design, operation, organization, and evaluation functions, (2) it is able to relate the abstract and concrete data elements of an interface, and (3) it enables knowledge-based systems to exploit the captured data.

Keywords

User interface languages, model-based user interface development, user-interface management systems, interface models

INTRODUCTION

The software industry is making a substantial effort to lay the foundation for a new computing model that will enable a standard way for applications to interoperate and interchange data. One of the basic building blocks for this model is the XML language. There is, however, a problem that the user interface software community faces as this new computing model emerges. A standardization effort has not yet emerged for representing and manipulating *interaction data*—the data that defines and relates all the relevant elements of a user interface.

In this paper, we propose a solution for the representation and manipulation of interaction data. We introduce XIML (eXtensible Interface Markup Language), an XML-based language that enables a framework for the definition and interrelation of interaction data items. As such, XIML can provide a standard mechanism for applications and tools to interchange interaction data and to interoperate within integrated user-interface engineering processes, from design, to operation, to evaluation.

XIML REQUIREMENTS

In order to effectively define a representation mechanism for interaction data, it is necessary to clearly establish the requirements of such a representation in terms of expressiveness, scope, and underlying support technologies. These requirements are:

- **Central repository of data**. The language must enable a comprehensive, structured storage mechanism for interaction data.
- **Comprehensive lifecycle support**. The language must enable support functionality throughout the complete lifecycle of a user interface.
- **Abstract and concrete elements.** XIML must be able to represent the abstract aspects of a user interface, such as the context in which interaction takes place, and the concrete aspects, such as the specific widgets that are to be displayed on a screen.
- **Relational support.** The language must be able to effectively relate the various elements captured within the scope of its representation.
- **Underlying technology.** In order to be useful within an industry-based new computing model, XIML must adhere to at least two implementation requirements**.** First is the use of an underlying technology that is compatible with that computing model. In this case, this points to the use of XML. Second, the language must not impose any particular methodologies or tools on the design, operation, and evaluation of user interfaces.

THE STRUCTURE OF XIML

The XIML language draws mainly from two foundations. One is the study of ontologies and their representation, and the other one is the work on interface models [2]. From the former, XIML draws the representation principles it uses; from the latter it derives the types and nature of interaction data. The XIML language includes the following representational units:

Components

In its most basic sense, XIML is an organized collection of interface *elements* that are categorized into one or more major interface *components*. These components are those typically found in an interface model [2]: user tasks, domain objects, user types, presentation elements, and dialog elements.

Relations

A *relation* in XIML is a definition or a statement that links any two or more XIML elements either within one component or across components. By capturing relations in an explicit manner, XIML creates a body of knowledge that can support knowledge-based design, operation, and evaluation functions for user interfaces.

Attributes

In XIML, *attributes* are features or properties of elements that can be assigned a *value*. The value of an attribute can be one of a basic set of data types or it can be an instance of another existing element.

VALIDATION OF XIML

In order to validate the expressiveness and usefulness of XIML, we undertook a number of tests and projects. These activities have the main goal of allowing us to assess the feasibility of XIML satisfying the requirements that we elicited for the language. The validation activities included among others:

Hand Coded Interface Definition

It is useful with any new language schema to hand code a few real-world target samples. This allows language designers to ascertain the range of expressiveness of the language as well as its verbosity—the size and number of expressions that would be necessary to code one example. Our hand coded specifications ranged from a single domain component to describe the catalog items of a store, to a task model for a supply-chain management application, to presentation components for simple C++ interface controls for Windows, to entire interface definitions for a number of applications (a geographical data visualization application, a baseball box-score keeper, and a dictionary-based search tool among others). In all of these examples, we found XIML to be sufficiently expressive to capture the relevant interaction data.

Multi-Platform Interface Development

One of the important uses of XIML can be in the development of user interfaces that must be displayed in a variety of devices. XIML can be used to effectively display a single interface definition on any number of target devices. This is made possible by the strict separation that XIML makes between the definition of a user interface and the *rendering* of that interface—the actual display of the interface on a target device. In the XIML framework, the definition of the interface is the actual XIML specification and the rendering of the interface is left up to the target device to handle. We have reported in detail elsewhere on an intelligent system that can automate to a large degree the development of interfaces for multiple devices using XIML [1]. We have shown in that work an example of displaying a map-annotation user interface on a desktop, a PDA, and a cell phone.

Intelligent Interaction Management

One of the main goals of XIML is to provide a resource for the management of a user interface at runtime. By centralizing in a single definition the interaction data of an interface, it is hoped that we can build tools that will similarly centralize a range of functions related to the operation of that interface. We have built prototype implementations that support, among others, the following runtime functions:

Dynamic presentation reorganization. XIML can support the reconfiguration of the layout of a user interface based on knowledge about the user interface that is also captured by the XIML representation. For example, the widgets used in the interface can be automatically adapted to the screen area available, or to the type of data being displayed.

Personalization. The interaction data captured by XIML can be exploited to provide a number or personalization functions, not only at the content level (as is already done by some commercial applications) but also at the interface layout and navigation levels.

Distributed interface management. One of the drawbacks of any client-based software application is that the update of the client software is problematic since each individual client needs to be updated. XIML provides a mechanism for the distributed update of user interface components. We have taken advantage of that flexibility to allow the widget to simply be available somewhere on the network be it on a client, a peer, or a server machine. The XIML specification can be set to link to providers of the widget or it can rely on a search-and-supply application. In this manner, for example, a calendar widget on a travel-reservations page can be provided by any number of XIML-compliant calendar-widget suppliers.

THE XIML FORUM

The task of defining, validating, and disseminating XIML for its adoption and standardization will be conducted through the XIML Forum (www.ximl.org). This organization will define a road map that ensures substantial participation by industry and academia in the development path of XIML.

REFERENCES

1. Eisenstein, J., Vanderdonckt, J. and Puerta, A. “Applying Model-Based Techniques to the Development of UIs for Mobile Computers”. In *IUI01: 2001 International Conference on Intelligent User Interfaces*. Santa Fe, NW, pp. 69-76.

2. Puerta, A.R. “A Model-Based Interface Development Environment”. In *IEEE Software*. 1997. pp. 40-47.