A Device-Independent Multimodal Mark-up Language*

Steffen Göbel\textsuperscript{2}, Falk Hartmann\textsuperscript{1,2}, Kay Kadner\textsuperscript{1,2}, Christoph Pohl\textsuperscript{2}

\textsuperscript{1}Department of Computer Science  \hspace{1cm} \textsuperscript{2}SAP Research CEC Dresden
Technische Universität Dresden  \hspace{1cm} SAP AG
Dresden, Germany  \hspace{1cm} Dresden, Germany

\{falk.hartmann, kay.kadner\}@tu-dresden.de
\{steffen.goebel, falk.hartmann, kay.kadner, christoph.pohl\}@sap.com

Abstract: Today’s growing heterogeneity of end user devices makes it crucial for application developers to deploy applications on as many devices as possible with an acceptable effort. Furthermore, new modalities like speech and gestures allow for more natural interaction especially with small mobile devices. It is hard to simply extend existing Web applications to support multimodal interactions. Hence, it is necessary to model modality-specific aspects in applications and their user interface description languages respectively. The Device-Independent MultiModal Mark-up Language (D3ML) addresses these issues and is described in this paper.

1 Introduction

A couple of trends have recently influenced the development of user interface (UI) description languages. Thin clients and browser-based information access is the currently prevailing paradigm for online information access. Mark-up languages like [HTM05] provide the means for easy authoring of rich user interfaces in a modular fashion.

The “Ubiquitous Web” requires not only the device-independent presentation of information taking into account different screen sizes ([DI01]) but also the integration of arbitrary input and output devices with a wide range of different capabilities, enabling multimodal interactions ([MMI02]). Potential use cases are plenty. Multimodal interactions do not only provide more natural ways to communicate with computers, they also open new accessibility options for people with special needs, like impaired users or workers that need to operate hands-free for performing other tasks.

The SNOW project (Services for NOmadic Workers) [SNO04, BPBH06] takes up these trends to provide a system environment with multimodal interactions on mobile devices for online access to structured maintenance documentation and for collaboration with remote experts in the context of aircraft maintenance. Part of these efforts is the definition

\*Work on this paper has been partially funded by the EU in FP6 IST SO 2.3.2.6, see [SNO04].
of a domain-specific language for describing Web-based user interfaces with the aspects of device-independence and multimodality. This means that abstract user interfaces modelled using our Device-Independent MultiModal Mark-up Language (D3ML) can be transformed to concrete UIs with any combination of available input and output modalities.

This paper is organised as follows: Sect. 2 forms the main part of the paper introducing the concepts of D3ML with respect to both processing and language design. Sect. 3 explains the validation of these concepts in the context of the SNOW project. Finally, we review a selection of related work with respect to UI description languages in Sect. 4 and conclude with a short outlook in Sect. 5.

2 Concepts of D3ML

D3ML is an abstract, Web-based user interface description language primarily intended for adaptation to various concrete mark-up languages, which can be directly rendered by browsers and comparable client applications. Adaptation ensures the dynamic reaction to changing device capabilities and user preferences. In the SNOW project there are considerations about extending existing browsers to directly interpret D3ML content. This is achievable due to the strict reuse of XHTML modules in D3ML. The following subsections will elaborate the various concepts implemented in the language design of D3ML. The specification of D3ML can be downloaded at [SNO04].

Processing model of D3ML. The processing model of D3ML is depicted in Fig. 1. In response to a specific user request, the backend application fills placeholders in D3ML templates with dynamic data from a backend database. The dialogue manager inserts navigational links between pages according to its configured dialogue flow model (e.g., login, index, search, details, help pages, etc.). Most important, the adaptation manager keeps track of the current execution context in terms of device characteristics and user preferences to adapt generic D3ML models to content formats processable by end user devices, like (X)HTML, SSML or VoiceXML. It can also decide to paginate content (i.e., split into sub-pages) to allow rendering on smaller screens or to split audio into chunks that are easier to keep in human short term memory. It can thus be seen as a dynamic, domain-specific model transformer. However, a specialised browser could also directly interpret D3ML similar to current multimodal browsers’ X+V (XHTML+VoiceXML, [Voi04]) rendering capabilities.

![Figure 1: Processing model of D3ML](image-url)
**Interaction model of D3ML.** In contrast to traditional browser-based Web applications, D3ML assumes a more detailed interaction model as depicted in Fig. 2. According to the architectural layer where they are handled, we distinguish three kinds of commands: *Application level commands* require navigation of the dialogue flow model or the invocation of the backend application to retrieve a new D3ML page. They are handled by the dialogue manager. *Page navigation commands* refer to additional navigation introduced by pagination at the level of the adaptation manager. *Browsing control commands* refer to simple user interface interactions, like scrolling or switching input focus etc.

![Diagram of Interaction model of D3ML](image)

Figure 2: Interaction model of D3ML

**Language foundations of D3ML.** D3ML was designed as a domain-specific language for modeling device-independence and multimodality in Web-based user interface descriptions. It is based on XHTML 2 as host language, which provides an extensible skeleton of a document as defined in XHTML Modularization ([HTM05]). D3ML consists of a set of integrated XHTML modules (i.e., XForms 1.1 REC for input validation, DISelect WD content selection, and XML Events REC command binding and event handling).

The combination of modules in D3ML reuses many concepts of the Renderer Independent Markup Language (RIML) [RIM04, SZG+03, ZLH04], developed in the EU IST project CONSENSUS with participation of SAP. The W3C Device Independence Working Group has also integrated many of these concepts in DIAL [DI01]. Yet, RIML was merely intended for the device independent description of UIs and not for multimodal interactions. It has several features to achieve this: classification of devices and layout containers for device classes, direct content control and support of device-native content, as well as pagination for splitting documents into renderable chunks satisfying usability requirements and author's intentions [SZG+03]. The one-dimensional device classification of RIML is insufficient for constructing UIs adaptable to both input and output modalities. This prevents, e.g., the consideration of multiple input capabilities for the same device class in different contexts. Hence we extended the concepts of RIML in D3ML with respect to multimodality. This decision also facilitates reuse of software and tools developed for RIML.
Content selection. Unfortunately, RIML introduced the concept of layout containers only for spatial arrangement of document content. A multimodal UI description language needs additional means for describing sequential arrangements of document content, e.g., for rendering non-spatial, time-based modalities like voice output. We have decided to permit multiple layout containers per D3ML document to allow the definition of sequential as well as spatial arrangements in one D3ML instance.

Content sharing between multiple layout containers helps keeping documents maintainable, small, and fast to transmit. In contrast to RIML, where layout containers may reference content from each other, D3ML layout containers are defined separate from content as meta-information in the XHTML head part. This cleanly separates layout from content and leaves the body structure of XHTML intact, which greatly simplifies authoring, previewing, and rendering of D3ML documents, as well as reusing XHTML content.

D3ML allows adaptation managers to select layout containers depending on several criteria by incorporating features from the DISelect proposal [DI01]. Available expressions include: access to device capabilities (e.g., availability of certain modalities), modality capabilities (e.g., ease of use), and user preferences (e.g., languages). This selection may yield multiple layout containers. Their synchronisation is discussed below.

Semantic enrichment of content. Besides structural adaptation, content adaptation plays a crucial role in creating device-independent and multimodal applications. To assist the adaptation manager in producing user-friendly output, D3ML content can be enriched with semantic meta-information—so-called rendering hints—using the XHTML Metadata/attributes modules. This semantic information must be provided by the author, because it cannot be extracted automatically from the document. Examples include: importance of sections, scalability of images on small screens, alternative representations for various modalities, semantic relations between sections (detailedBy). The supported set of meta-information is not limited by D3ML. It depends on the capabilities of the adaptation manager.

Command binding and event handling. All important parts of an application should be conveniently accessible, no matter which modalities are used. D3ML allows defining document-actions to support this behaviour. Actions are modality independent, but can be augmented with modality-specific constraints. The concept of mapping modality-specific user commands to document-specific and modality-independent actions is called command binding. This feature can be used for permitting/prohibiting certain input modalities.

In analogy to XHTML 2, the XML Events module is used for command binding in D3ML. Event handlers in the head section map recognized commands (speaking, clicking, gesturing, etc.) to actions, which are either URIs defining subsequent documents (page navigation or application level commands) or a browser commands like scroll up. Global commands can also be defined in this manner. Thus, the behaviour of a document is defined in central place.
Synchronisation between layout containers  If multiple layout containers are selected for simultaneous rendering (e.g., voice and visual UI in parallel), their state must be synchronised to ensure a consistent view of the application. This synchronisation is realised by referring to the same content sections from each layout container via unique IDs. The adaptation manager (Fig. 1) has to ensure that sections with equal IDs are rendered at the same time.

![Diagram of layout containers]

The document in Fig. 3 has a headline area (A), a navigation area (E), and content area (B,C,D). It defines two layout containers for spatial (e.g., visual) and sequential (e.g., audible) ordering of the contained blocks. Note that the sequential layout has a different ordering of the content blocks (B,D,C instead of B,C,D in the spatial container), e.g., for better comprehensibility of the audio representation. The upper branch shows a bad pagination example: The user will be confused if he sees something he cannot hear or vice versa; a good pagination must preserve the order and the correlation between paginated parts of multiple modalities.

Example  In Listing 1, the skeleton of a very simple D3ML file is shown. Most of the namespace declarations and some other details have been removed to improve readability.

Starting in line 3, this example file contains a layout declaration that is used as the basis for the feature of content sharing as described above. A second layout container for audible output is not shown in detail but might refer to the same content section sec0.

Beginning in line 17, a handler and a listener (in the XML Events sense) are defined that refer to the link l1 and the event e1.

The link l1 is defined in the body of this D3ML document in line 24. The link is attributed with meta-information that assigns the event e1 to the recognition of the word Removal by the speech recognition.

Listing 1: Basic D3ML example

```html
<html ... xmlns:d3ml="http://www.snow-project.org/2005/11/d3ml">
  <head>
    <d3ml:layout>
      <sel:select>
        <sel:when expr="outputMethod() = ‘visual’”>
          
```
3 Validation

As mentioned introductorily, D3ML is currently being deployed in the mobile worker environment of the SNOW project ([SNO04]) with the primary use case of multimodal online access to maintenance documentation in harsh environments like aircraft maintenance, repair and overhaul. Modalities under current consideration include traditional visual output and stylus-/keyboard-based input, speech recognition and text to speech to allow hands-free operations, as well as gesture input as a fall-back when background noise denies reliable speech recognition. The architecture chosen for the SNOW project closely resembles the multimodal interaction framework proposed in [MMI02]. More details of this architecture are described in [BPBH06].

With respect to the processing model (Fig. 1) and interaction model (Fig. 2) of D3ML, documentation application, dialogue manager, and adaptation manager are implemented using current Web application framework technology, like Java Servlets and Apache Cocoon, among others. The initial mobile client application is an augmented browser running on Pocket PCs and Tablet PCs. The assessment of first integration tests has shown the general feasibility of the presented concepts.
4 Related Work

Several other user interface description languages have been developed to address one or both aspects of device-independence and multimodality on varying levels of abstraction. Web-based approaches like X+V (XHTML+Voice Profile, [Voi04]) and SALT (Speech Application Language Tags, [SAL02]) tackle the problem at the level of final user interfaces. X+V uses two dialogue models for the same application—an XHTML body for visual elements and VoiceXML in the header for speech I/O—and connects their events via ECMAScript. The overlap between both dialogue models implies unnecessary redundancy and it is hard to integrate further modalities like gesture recognition. SALT injects a set of defined XML tags and attributes into the source document syntax of HTML or other mark-up languages. Although the visual and audile dialogue models are tighter interwoven than with X+V, the general approach is the same and thus also the differentiation against D3ML. In contrast, the main idea of D3ML is to have only one unified dialogue model for all possible modalities. As described in Sect. 2, alternative layouts and additional metadata allow for easy adaptation to new I/O capabilities and user requirements. This degree of device-independence is neither covered by X+V nor by SALT.

While D3ML has a slightly higher abstraction level than Web-based multimodal mark-up languages X+V and SALT, other approaches go even further by modelling abstract user interfaces or even generic task concepts. UsiXML (USer Interface eXtensible Markup Language, [LVM+04, VLM+04]), an XML-based mark-up language to describe UIs for multiple contexts of use, such as Character User Interfaces (CUIs), Graphical User Interfaces (GUIs), Auditory User Interfaces, and Multimodal User Interfaces, describes user interfaces at decreasing levels of device abstraction: Tasks & Concepts, Abstract User Interface, Concrete User Interface, and Final User Interface. Graph transformation techniques and graph grammars are used to formalise inter-model mappings consisting of abstractions, reifications and translations. UsiXML follows a completely different approach of UI design compared to our D3ML or traditional web application developments. Thus, even with reverse engineering techniques reusing existing Web content in UsiXML is difficult, in contrast to D3ML. Additionally, the learning curve for UsiXML is higher than for D3ML considering developers with HTML knowledge.

5 Conclusion

In this paper we have proposed the Device-Independent MultiModal Mark-up Language (D3ML) as a general-purpose description language for Web-based user interfaces allowing developers to model enough meta-information for adapting output to any useful combination of input and output modalities. D3ML provides a number of important features: multiple layout containers for alternative modality-dependent arrangements of UI elements, aggregation of all meta-information relevant for rendering content on arbitrary devices and/or modalities, synchronisation of multimodal I/O using IDs for UI elements, flexible
command binding and event handling concept, modular integration of new concepts into existing standards, and easy reuse of existing XHTML-based content.

Initial results of the validation presented in Sect. 3 are promising. Feedback gathered from these tests will influence the further development of D3ML. D3ML has been submitted to the W3C Multimodal Interactions Working Group ([MMI02]) as input for their standardisation activities. Furthermore, an authoring environment for D3ML-based Web applications is currently being implemented in the SNOW project [SNO04].

References


