Watermarking Integration into Portals

Patrick Wolf
Fraunhofer IPSI, Germany

Martin Steinebach
Fraunhofer IPSI, Germany

INTRODUCTION

Digital watermarking has become an accepted security technology to protect media such as images, audio, video, 3-D, or even text-based documents (Cox & Miller, 2002). Watermarking algorithms embed information into media data by imperceptible changes of the media. They enable copyright or integrity protection, broadcast monitoring, and various other applications. Depending on targeted application and media type, various concepts and approaches for digital watermarking exist.

MOTIVATION

So far, watermarking algorithms were encapsulated in individual applications. For example, before customers download an audio book from an online store, the audio book gets watermarked with the customer ID, thus, individualizing the audio book allowing tracing of illegitimate publications back to the original customer (Steinebach & Zmudzinski, 2004). Usually, such applications only deal with a single type of media, and the algorithm is tightly integrated into the application's workflow. What is missing is the flexibility to watermark a medium of arbitrary type with a number of appropriate watermarking algorithms.

On Web sites of watermarking algorithm developers, one can often find samples of watermarked media to prove the watermarks imperceptibility. But still, when dealing with interested persons that want to use watermarking technology, the most common question from them is: "Does the watermarking process degrade the quality of *my* media?" The best answer to this is to have them apply the watermarking to *their* media. But for this, in many cases, a direct contact between the interested person and the watermarking developer is necessary, since the appropriate algorithm and the best configurations have to be determined. Therefore, a simple way of allowing everyone to test watermarking using their own media, whatever they are, would greatly facilitate the acceptance of watermarking.

Portals are the ideal technology for this scenario. A watermarking portal could bring together potential users and algorithm developers. Service functionality, like file upload and graphical information exchange, have long been in place. So such a portal could concentrate on the crucial issues of the watermarking workflow. Alas (as described), watermarking algorithms tend to be not flexible or generic enough to allow watermarking of arbitrary media. This is due to the fact that for watermarking a certain media type, it is necessary to deal with the semantics of this media type and the semantics of, for example, an image and an audio file vary highly.

But still, there is a lot that watermarking algorithms have in common. When these common issues are refined into uniform interfaces, a watermarking portal could be built where developers register their compliant algorithms and the portal offers them to users, enabling them to watermark digital media of arbitrary types.

This article is structured as follows. After a short introduction to the basic concepts of digital watermarking, we will describe a set of challenges one encounters when integrating watermarking as a technology into Web applications using the Fraunhofer Watermarking-Portal as an example. The article is concluded by a summary and some remarks on future trends.

DIGITAL WATERMARKING

Digital watermarking describes the process of attaching information inseparably to a digital medium such as image, audio, video, or text to provide some form of added-value (Sequeira & Kundur, 2001). The attachment of information is known as watermark embedding and the attached information as message (sometimes also simply *the watermark*). Watermarking algorithms usually slightly alter certain characteristics of the carrier medium (like the relation of energy of frequency bands in audio or average brightness of pixel-blocks in images), so that afterwards they represent the embedded information. Watermark retrieval (or detection) algorithms try to read the embedded information. In order to ensure security and confidentiality, the message is often protected by a secret key and without its knowledge, it is hardly possible to access, alter, or remove the message from the medium.

Watermarking is widely described as a form of communication (Cox, Miller, & McKellips, 1999), with the sender embedding the message into a carrier signal (the medium) and the receiver retrieving the message. From this point of view, watermarking algorithms have important characteristics:

Imperceptibility describes how much (or rather how little) the embedding process perceivably changes the carrier medium. *Capacity* describes the number of bits that can be transmitted through the watermarking process. *Robustness* measures how stable the embedded information is against alterations of the carrier medium. Finally, *security* describes how secure the embedded information is; without the knowledge of a secret key, the embedded information should neither be accessible, alterable, or removable. There are important differences between security of classic cryptographic systems and watermarking systems. A prominent aspect is the fact that attackers without control over a watermark detector can never be sure if they have removed the embedded information (Cayre, Fontaine, & Furon, 2005).

It is important to note that, in contrast to any other form of enriching the medium with information, the product of the embedding process is still a digital medium that can be consumed, transferred, and processed without restrictions.

Applications for digital watermarking range from copyright and integrity protection via broadcast monitoring to simple annotation. For copyright protection, robust watermarking algorithms are used that either identify sender (copyright holder) or receiver (buyer) of a digital medium. Integrity protection uses both fragile algorithms (embedded information is destroyed when medium is altered) and robust algorithms (robustly embedding important features of the original medium). For broadcast monitoring and annotation watermarking, robustness constraints are not so severe; algorithms with high capacity are needed in this situation.

The range of applications and the interplay of algorithm characteristics involved give a hint how complex selection of appropriate algorithms is. A portal as an intermediary between algorithm developers and potential users might lower entrance boundaries. This moves the need for selection and parameterization of algorithms from the users to the experts integrating the watermarking technology into the portal (Thiemert, Steinebach, Dittmann, & Lang, 2003).

A PORTAL APPROACH TO WATERMARKING

This section describes challenges that arise when integrating watermarking technology into portals, and their resolutions. In order to have a more graspable background, we will illustrate challenges and resolutions using an existing portal, the Fraunhofer Watermarking Portal (Fraunhofer IPSI, 2006), a Web application for watermarking personal media. The solutions presented can easily be generalized and most are applicable to any portal trying to integrate watermarking technology.

Each portal has two portal aspects. From a consumer side, it is a portal to watermarking algorithms allowing watermarking of their media. From an algorithm developer side, it is a portal for watermarking algorithms in the sense that arbitrary algorithms (independent of media type or implementation language) should be registrable.

CHALLENGE: GENERIC STRUCTURE

All Web applications, including portals, should have an internal structure that is as generic as possible. This is a rather universal prerequisite that applies, especially in fields that are not consolidated yet, standards are not yet established, and the full impact of the technology cannot be foreseen. Watermarking is such a field. A portal trying to integrate watermarking should therefore be able to react as flexibly as possible to the changes that will surely come. Such changes include, but are not limited to, watermarking algorithms, media processing, and analysis or production workflows.

Resolutions to this challenge are multitude. One established answer to the challenge is the model-view-controller (MVC) paradigm (Burbeck, 1992). Many Web application frameworks have been upon this paradigm; Struts being the most prominent example (Apache Struts). The larger and more powerful the framework, the more complicated its configuration can get. This is not ideal for integrating watermarking. We therefore propose a more lightweight variant customized for a finer control of the request/response flow.

An HttpServlet within a Tomcat application server serves as the FrontController of the Web application accepting the client's (usually a Web browser) request, and selecting and assembling the response. The Views are JavaServer-pages (JSP) and the Model consists of a large collection of different classes including classes responsible for representing and storing media as well as algorithms. Any interaction with the Web application is modeled as a *Process* that consists of *Steps*. A login-process might consist of a *Step* that displays a login form, which is submitted and validated. Supplying correct credentials lead to an activation *Step* (after the very first login) or to the welcome *Step*. Failing to do so, leads back to the *Step* that displays the login form. In contrast to Struts, Views have no knowledge of the next *Steps* and thus, only request the next *Step* of a *Process* (actually, this is also possible in Struts, but leads to a single "action" with a complex "forward" hierarchy). Each *Step* has Views assigned to it. Prior and subsequent to displaying Views, *WebActions* can be performed. They are named *Pre*- and *PostStepActions,* accordingly. It is the *WebActions* that actually define what the application is doing. They define its semantics while

Related Content