

# WAPcam - eine multimediale WAP Anwendung in der Studentenausbildung

Frank Kargl (frank.kargl@informatik.uni-ulm.de)  
Torsten Illmann (torsten.illmann@informatik.uni-ulm.de)  
Alexander Raschke (alexander.raschke@informatik.uni-ulm.de)  
Stefan Schlott (stefan.schlott@informatik.uni-ulm.de)

Universität Ulm  
Abteilung Medieninformatik

17.01.2001

## 1 Einleitung

Im Rahmen eines geplanten Praktikums mit dem Titel “Web-Engineering” entstand im Frühjahr 2000 an unserem Lehrstuhl die Idee, auch das Themengebiet WAP aufzugreifen. Das Praktikum selbst stellt ein Ergänzungsangebot zu einer gleichnamigen Vorlesung dar und vertieft in Themen wie Dokumenten- und Layoutsprachen im Web, Server- und Klienten-seitige Applikationsentwicklung fürs Web, Interaktionsformen und vieles mehr.

Von unseren Kollegen am TECO in Karlsruhe, mit denen sowohl die Vorlesung als auch das Praktikum in Kooperation entwickelt konzipiert und weiterentwickelt wird, übernahmen wir die Idee, auch WAP und insbesondere dem Thema “Messen/Steuern/Regeln mit Web und WAP” einen eigenen Platz einzuräumen. Während im vorausgegangenen Semester in Karlsruhe das Praktikum sich mit einfachen Schalt- und Regelvorgängen wie dem Ein- und Ausschalten einzelner Dioden beschäftigte, wollten wir allerdings einen Schritt weitergehen. Unsere Idee war es, eine interaktiv steuerbare Kamera über WAP zugänglich zu machen: die *WAPcam*.

## 2 Anforderungen

Bevor wir uns an eine Umsetzung heran wagten, wollten wir zunächst klären, unter welchen Rahmenbedingungen ein solches Unternehmen realisiert werden

konnte.

Aus technischer Sicht war zunächst zu definieren, welche Funktionalität die Anwendung haben sollte. Die Kamera sollte dreh- und schwenkbar gelagert sein und sich über das Handy intuitiv bedienen lassen. Wegen des kleinen Displays sollte weiterhin eine Zoom-Funktion integriert sein, um Details besser erkennen zu können. Da die Displays bei den gängigen Mobiltelefonen keine Graustufen- oder gar Farbdarstellung unterstützen, war weiterhin eine Kontrast- bzw. Helligkeitsregulierung vorgesehen.

Neben den technischen Aspekten, die unmittelbar für die Implementierung eines Prototypen interessant waren, mussten wir selbstverständlich auch die organisatorischen Aspekte im Praktikum berücksichtigen. Schon aus Kostengründen war klar, dass die Studenten primär mittels eines WAP Simulators [1] entwickeln und testen würden. Das Praktikum beinhaltete bereits eine Vielzahl unterschiedlichster Technologien (HTML, CGI, Perl, PHP, Java, Servlets, Applets, XML, XSL, ...), die den Studenten zumindest in der praktische Anwendung teilweise völlig neu waren. Von daher war der Arbeitsaufwand nicht zu unterschätzen und wir hatten zunächst Bedenken, einen weiteren, komplett neuen Technologiebereich (WAP, WML [2], WMLS [3]) mit einzubinden. Wenn überhaupt, dann musste diese Aufgabe, die am Semesterende auch den letzten Aufgabenbereich bilden sollte, mit Sicherheit entsprechend spannend und ansprechend gestaltet werden, um die Motivation deutlich über die Aussicht auf einen Praktikumsschein zu heben. Wir wollten also in jedem Fall neben der Simulationsumgebung auch die Arbeit an richtigen Mobiltelefonen ermöglichen. Auf unsere Anfrage hin stellte uns das Entwicklungszentrum der Firma Siemens in Ulm ausreichend wap-fähige Geräte des Typs Siemens S35i zur Verfügung. Die zentrale Universitätsverwaltung versorgte uns mit notwendigen SIM-Karten, ohne die ein Mobiltelefon nur eine nutzlose Hülle bleibt. Um die Motivation weiter zu steigern, bauten wir um die Kamera herum eine Art Marslandschaft auf, die ein verstecktes Rätsel enthielt, welches es zu entdecken und zu enträtseln galt.

### 3 Systemarchitektur

Der grobe Aufbau der WAPcam Steuerung wird in Abbildung 1 dargestellt. Die Kamera ist dabei auf einen dreh- und schwenkbaren Fuß [4] montiert. Das Auslesen der Kamera sowie die Steuerung des Kamerafusses erfolgt über einen separaten Kontrollrechner. Dieser stellt die Funktionalität über einen Java RMI Dienst [5], welcher einfach von anderen Rechnern angesprochen werden kann, zur Verfügung. Die RMI Schnittstelle beinhaltet Methoden, um den Kameraarm entweder absolut auf bestimmte Winkel zu positionieren oder aber relativ um einen Winkel zu drehen und zu schwenken. Darüber hinaus läßt sich der Kamerafuß in die Ausgangsstellung zurücksetzen und es kann ein einzelnes Kamerabild ausgelesen werden. Die RMI Schnittstelle wurde von uns entwickelt und den Studenten zusammen mit den zugehörigen Stub-Klassen und Testdateien zur Verfügung gestellt.

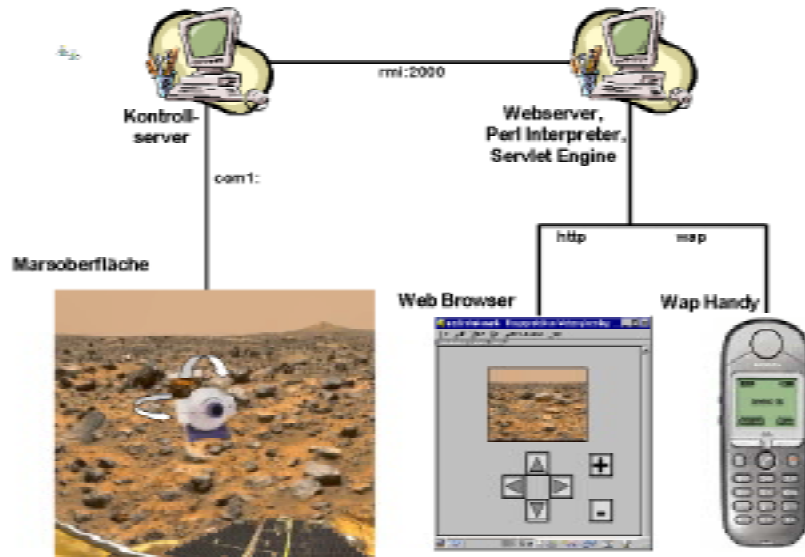


Abbildung 1: Aufbau der WAPcam Steuerung

Um verwirrendes Verhalten der Kamerasteuerung bei konkurrierendem Zugriff aus mehreren Anwendungen zu vermeiden, ist eine explizite An- und Abmeldung notwendig. Bei der Anmeldung erhält die Anwendung ein Token, welches nach der Abmeldung oder nach einem Timeout seine Gültigkeit verliert. Auf einem weiteren Rechner ist ein Webserver installiert, der den Zugriff via HTTP ermöglicht. Bei WAP ist es wegen der eingeschränkten Endgeräte und den beschränkten Bandbreite sinnvoll, soviel Verarbeitungsschritte wie möglich lokal im Server durchzuführen und das Endgerät nur zum Anzeigen der Daten und für einfachste Operationen in Anspruch zu nehmen. Die Anwendungslogik ist daher weitestgehend im Webserver in Form eines CGI-Skripts [6] oder Servlets [7] realisiert. Folgende Funktionen sind umzusetzen:

- Bewegungssteuerung der Kamera gemäß den Steuerbefehlen des Endgerätes und
- Abrufen der Bildinformation und (*sinnvolle*) Umwandlung in ein WBMP Bild [8].

Insbesondere die Bildwandlung verdient eine besondere Berücksichtigung. Das von der Kamera (über den RMI Dienst) gelieferte Ausgangsbild hat eine Abmessung von 640x480 Bildpunkten und ist JPEG kodiert. Beim Empfänger

kann lediglich ein schwarz-weiss Bild mit etwa 100x70 Bildpunkten dargestellt werden, wobei diese Größe noch starken Schwankungen unterliegt. Im Vorfeld äußerten mehrere Teilnehmer ernsthafte Bedenken, ob unter solchen Voraussetzungen überhaupt ein sinnvoller Einsatz dieser Technologie möglich ist. Dazu werden wir im letzten Kapitel Überlegungen anstellen. Es ist jedoch offensichtlich, dass bei der Konvertierung die Bildinformation aufbereitet werden muss. In unserer aktuellen Implementierung verwenden wir folgende Bildbearbeitungsfunktionen, um eine sinnvolle Umwandlung in ein WBMP Bildformat herzustellen::

1. Umwandlung in ein Graustufenbild,
2. Kanten- und Kontrastverstärkung,
3. Ausschnitt (dies realisiert die gewünschte Zoom-Funktion),
4. Skalierung auf das gewünschte Zielformat und
5. Umwandlung in Schwarz/Weiss mit Floyd-Steinberg Dithering (mit einstellbarem Threshold: dies realisiert die gewünschte Helligkeitseinstellung).

Dabei ist es nicht ganz einfach, das Zielformat zu ermitteln. Manche WAP Browser übertragen zwar die maximale Displaygröße oder den Typ des Mobiltelefons, leider ist dieser Mechanismus nicht standardisiert, so dass man oft auf den kleinsten gemeinsamen Nenner zurückgreifen muss und daher die Möglichkeiten des Endgeräts unter Umständen nur ungenügend ausnutzt. Hier sollten im Standard entsprechende Erweiterungen vorgenommen werden.

Die Funktion des WAP Gateways haben wir nicht selbst realisiert. Wir verwenden hier vielmehr die WAP Gateways der jeweiligen Netzbetreiber. Mit den uns zur Verfügung stehenden Karten war der Zugriff auf die Gateways von D1 und D2 möglich. Hiermit traten während der Tests keine Probleme auf, von einer häufigen Nicht-Erreichbarkeit einmal abgesehen.

## 4 Prototypische Implementierung

Im Vorfeld des Praktikums wurde das System prototypisch implementiert, um die Machbarkeit zu demonstrieren. Das Bild wird in eine statische WML Seite eingebettet und von einem Perl CGI-Skript dargestellt. Steuerungskommandos, Zoom-Faktor und Helligkeitsswerte erhält das CGI-Skript über Parameter mitgeteilt. Da wir zur Bedienung des Systems eine (relativ) grosse Anzahl von Tasten benötigen und uns daher die wenigen Softkeys nicht ausreichen, haben wir uns entschlossen, die Zifferntasten des Mobiltelefons zu belegen. Dabei entsprechen die Tasten 2, 4, 6 und 8 den Richtungen hoch, links, rechts und runter. Über die Tasten 1 und 3 ist die Zoom-Funktion zu steuern und 7 und 9 regeln die Helligkeit. Die Taste 5 setzt die Steuerung auf Ausgangsstellung zurück. Diese Lösung ist in der Praxis sehr gut bedienbar, hat aber leider zwei Nachteile:

- Die Zifferntasten lassen sich nur mit einer proprietären Erweiterung des von den Siemens-Mobiltelefonen verwendeten Browsers von Phone.com [1] belegen (ACCESSKEY Attribut von <ANCHOR>). Damit sind die Seiten nicht mehr allgemein übertragbar.
- Die ANCHOR Tags erscheinen als Textzeilen unter dem Bild. Nach Auslösen einer Funktion muss man u.U. manuell nach oben scrollen, um das Bild vollständig zu sehen.

Unserer Meinung nach bietet der heute implementierte WML/WMLS Standard noch nicht genügend Möglichkeiten für umfangreiche interaktive Applikationen. Daher ist zu befürchten, dass Entwickler verstärkt auf proprietäre Erweiterungen der Hersteller ausweichen werden. Hier sollte der Standard schnell so erweitert werden, dass zumindest alle sinnvollen Ein- und Ausgabemechanismen eines Mobiltelefons (und dazu gehören sicher die Zifferntasten) erfasst werden können.

Die Implementierung war relativ problemlos und lief im Simulator von Phone.com bereits recht früh. Allerdings zeigte sich bei den Mobiltelefonen ein unangenehmer Fehler. WML Variablen (die über <setvar> gesetzt bzw. in WMLS modifiziert wurden), werden bei den uns zur Verfügung stehenden Mobiltelefonen innerhalb von Attributwerten immer in einen leeren String evaluiert. Da somit keine dynamische Parameterübergabe an das Bildskript möglich war (kodierte in der URL des aufgerufenen CGI-Skriptes), mussten wir hier auf eine Ersatzlösung ausweichen. Dabei wird die WML Seite, die das Bild einbindet, dynamisch über ein weiteres Perl CGI-Skript erzeugt und die entsprechenden Parameter hart in die URLs kodiert.

Die Sourcen unserer Implementierung werden nach Ablauf des Praktikums Mitte Februar auf unserem Webserver (<http://webeng.informatik.uni-ulm.de>) abrufbar sein.

## 5 Erfahrungen und Anwendungsgebiete

Zunächst einige technische Aspekte. Obwohl die Idee der sinnvollen Realisierung einer interaktiven Kamera mit WAP bei vielen Leuten (auch Mobilfunkentwicklern!) auf grosse Skepsis stiess und bezweifelt wurde, liessen wir uns nicht von dem Versuch abbringen und wollten im schlimmsten Fall zumindest diese Skepsis mit Tests bestätigen. Die Ergebnisse des fertigen Prototyps stossen hingegen in der Regel auf eine positive Resonanz. Alles steht und fällt mit der Bildqualität. Mit den heute vorhandenen Endgeräten lassen sich je nach Motiv immerhin befriedigende bis gute Ergebnisse erzielen. Die zukünftige Entwicklung hin zu Geräten mit höherer Auflösung und Farbfähigkeit sollte die heutigen Qualitätsprobleme kurz- bis mittelfristig beseitigen.

Doch auch heute schon sind sinnvolle Anwendungen möglich. So liesse sich mit Hilfe von WAP ein privates Überwachungssystem realisieren, mit dem Hausbesitzer erkennen können, ob eingebrochen wurde oder Pflanzen gegossen werden müssen. Zusammen mit weiteren Steuerungsfunktionen (z.B. Jalousinen,

automatische Bewässerungsanlage usw.) könnte ein wirklicher Mehrwert geschaffen werden. Neben den eingeschränkten Endgeräten sind natürlich auch die geringe Übertragungsbandbreite und vor allem das starre und teure Preissystem der Mobilfunkunternehmen ein Hindernis bei der Durchsetzung solcher Anwendungen.

Unsere Studenten, die momentan die gestellte Aufgabe bearbeiten, nahmen die Aufgabenstellung jedenfalls sehr interessiert an. Die zahlreichen Kinderkrankheiten und Inkompatibilitäten, mit denen WAP heute noch zu kämpfen hat, sind sicher gerade in der Ausbildung problematisch, stören jedoch bei entsprechender Motivation der Teilnehmer nur geringfügig.

## Literatur

- [1] Phone.com: UP SDK Development Kit, Release 4.1, 2000.  
URL: <http://developer.phone.com/>
- [2] WAP Forum: Wireless Markup Language Specification, April 30, 1998.  
URL: <http://www.wapforum.org/>
- [3] WAP Forum: WMLScript Specification, April 30, 1998.  
URL: <http://www.wapforum.org/>
- [4] Directed Perceptions Inc: Pan-Tilt Tracking Mount - C Programmer's Interface, Version 1.0707b, 1995.
- [5] Sun Microsystems Inc: RMI Specification.  
URL: <http://java.sun.com/j2se/1.3/docs/guide/rmi/spec/rmiTOC.html>
- [6] E. Wilde: Wilde's WWW. Technical Foundations of the World Wide Web, Springer Verlag, 1999.
- [7] Sun Microsystems Inc: Java Servlets Specification, 2001.  
URL: <http://java.sun.com/products/servlet/2.2/>
- [8] WAP Forum: Wireless Application Protocol Wireless Application Environment Specification Version 1.1. URL: <http://www.wapforum.org/>