## Waterwave Webdesign

For the web projects made for Waterwave Webdesign I worked with PHP, XHTML, CSS, JavaScript and Mysql.

**redparty.ch** / September 2006 – November 2006
A community website for an annual party event. It has a custom Content Management System, a simple weblog and a photo gallery.
I implemented the PHP backend for the CMS, the weblog and the photo gallery as well as all the XHTML/CSS frontend (except the graphic design).

**leherbauer.com** / August 2005 – October 2005
A band website with a CMS a guestbook and a photo gallery.
I developed the CMS and guestbook in PHP and was responsible for the XHTML/CSS programming.

**lehrmittelbau.ch** / 2004
The presentation and online-store of a technical authors council.
I was responsible for developing the online-store and the XHTML/CSS programming.

**waterwave.ch/weblog** / since 2002
My personal weblog. It comes with a multilingual user interface, multilingual content handling, comments, trackbacks, user-selectable layouts/skins, XML-RPC API for remote posting, spam-protection, RSS- and atom feeds.
I developed it in PHP from scratch and did the XHTML/CSS and graphic design.

**Other websites made by Waterwave Webdesign** / 2001 - 2007
I did the PHP and XHTML/CSS/JavaScript development for all websites made by Waterwave Webdesign. This includes custom content management systems, photo galleries, news-management systems, restricted access member content and weblogs.
Websites: http://waterwave.ch/index.php?go=referenzende

## KIS, EPFL

**actualites.epfl.ch** / 2006 – 2007
actualites.epfl.ch is the news portal of EPFL. It contains several different types of news and press releases. My task was to create the front page of this news portal which should contain the latest news from all the different news sources but at the same time it should be possible to manually decide which news items should appear in which order and furthermore is should be possible to insert additional content boxes or to import content from external RSS feeds.
I created an administration interface where the content of the front page can be composed from the different news sources. To simplify this process and to improve the usability I added an AJAX based drag-and-drop interface. It includes a WYSIWYG editor to compose additional HTML content and to adapt imported content. There is also a global search function for the news sources and an import manager for RSS feeds. The application is designed modular and defines a simple API which makes it very easy to add additional news sources.
Product website: http://actualites.epfl.ch/

**inForm** / 2004 – 2006

inForm is an application to create webforms of any purpose without having to "programm" anything or know anything about HTML. It is used to create forms for inquiries, surveys, registrations, questionnaires, evaluations and votes.

A MySQL database is used to store the forms and the submitted data. InForm has optional LDAP based access control, has a multilingual interface, supports multilingual forms, creates graphical result charts, supports custom templates and styles, exports to Excel and CSV, provides access to the results through a XML-RPC API and integrates with the e-payment service of the swiss post.

I took inForm from prototype stage to a deployed application. During this I implemented LDAP access control, made the interface multilingual, added support for multilingual forms by providing a translation manager, transformed the templates and styles to valid XHTML and CSS, developed the Excel and CSV export functionality, fixed some performance issues, added an XML-RPC API, implemented access restrictions in order to allow the creation of confidential vote forms. I also progammed a sample XML-RPC client in Python which fetches the submitted data of a form from the inForm server and displays it in a wxPython GUI.

Product website: [http://inform.epfl.ch/](http://inform.epfl.ch/)

Source code: [http://inform.epfl.ch/inform-2.0.5.zip](http://inform.epfl.ch/inform-2.0.5.zip)

Documentation: [http://inform.epfl.ch/docs/](http://inform.epfl.ch/docs/)

**wiki.epfl.ch** – prototype / 2005

I developed a prototype for a custom EPFL wiki based on the MoinMoin wiki software.

For this I integrated a custom LDAP authentication module and added an administration interface to create multiple wikis and manage their access permissions through LDAP. Since MoinMoin showed some performance and storage issues, the prototype was canceled and a custom wiki software was developed based on the blogs.epfl.ch code.

## NovaWave Technologies

**DFG laser system** / March 2007 – September 2007

The DFG has an embedded Linux system with a touchscreen to control the laser parameters. The software is composed of two parts. A server which interacts with the hardware and provides a SOAP interface and a GUI which runs in a webbrowser and can be used through the touchscreen or from a PC on the network.

My first task was to do some performance analysis to find the cause of mysterious crashes. I used AWK to transform logfiles of resources usage into Gnuplot charts which showed that a memory-leak in the Mozilla JavaScript engine caused the crashes. To avoid this I transformed the XUL based GUI prototype into standard- and cross-browser compatible XHTML, CSS and JavaScript and used the Opera browser to replace Mozilla.

To read out an additional hardware sensor, I programmed some low-level C code and modified the affected SOAP API calls to consider this new sensor. I further consolidated and extended the C++ SOAP API of the server. During this I used Python to setup a unit test framework for the SOAP interface. This not only helped to discover some bugs but was also used to demonstrate some of the security holes of the server software. I found and fixed more security problems in the server and GUI parts of the software and introduced a limit mechanism to protect the laser from

dangerous parameter values.

By using AJAX to immediately show error messages or warnings for wrong parameter values and by simplifying the input of numerical values, I improved the usability of the GUI.

To fix some conflicts between the browser, the window manager and the touchscreen, I came up with a custom X11 window manager.

In addition to the development made on the DFG, I put the source code under version control, introduced a bugtracking system, added a documentation wiki and setup a Debian Linux server to run these applications.

Product website: http://www.novawavetech.com/products/products/instruments/iris-1000-dfg-based-mid-infrared-laser-system.html

Source code (only the source of the custom X11 window manager is publicly available):
http://svn.x-way.org/browse/listing.php?path=/x-wm/branches/nwt-dfg/

## School and personal projects

**Broadcom 43xx Linux Driver** / Summer 2005

I was part of the team who developed the Linux driver for the Broadcom bcm43xx wireless chip using reverse engineered specifications. It has been included in the Linux Kernel since 2.6.17-rc2. There was no Linux driver available before and Broadcom refused to provide specifications to the Linux developers. By owning a Apple PowerBook which uses this chip I was directly affected by the lack of a driver and therefore decided to join this project in order to make the wireless card work with Linux. I mostly worked on the implementation of low-level routines by transforming the specification into C code and less on high-level tasks like the integration into the Linux driver model.

Project website: http://bcm43xx.berlios.de/

Source code: http://git.kernel.org/?p=linux/kernel/git/torvalds/linux-2.6.git;a=tree;f=drivers/net/wireless/bcm43xx

**Clustering using task-allocation** / October 2006 – March 2007

Project of the Swarm Intelligence Course which had to be done in groups of two students. The goal was to develop a threshold-based/market-based algorithm to perform clustering using task-allocation and to implement this algorithm with a simulation environment to be able to gain performance results which can be compared to existing algorithms. The final Java application consists of a GUI to visualize the algorithms and a simulation framework where different algorithms can be compared.

I mostly worked on the simulation framework and the algorithm (esp. the "moving behavior" improvements).

Project report: http://files.waterwave.ch/dotta_jaggi_project_report.pdf

**Integrating new hardware components in GSN** / October 2006 – February 2007

Semester project which I did at the LSIR (Distributed Information Systems Laboratory). The goal was to integrate new hardware components in the Global Sensor Network (GSN) Framework, namely the Picotux, the BTnode and the NSLU2.

The Picotux is a very tiny embedded x86 Linux device containing a serial port and an ethernet port. I programmed it such that it allows GSN to access serial-port based sensors over the network

by forwarding the data from the serial port to the network.

The BTnode is a AVR-microprocessor based embedded device which comes with a couple of sensors (light, temperature, sound, movement) and has USB and bluetooth connectivity.

I developed an application in C which creates a bluetooth network between multiples BTnodes and sends all sensor values to a sink node which is connected to a PC where the data is put into GSN. To monitor the BTnodes I wrote a simple Java application which controls the battery levels of the BTnodes and displays their sensor values.

The NSLU2 is a consumer device running x86 Linux and is equipped with two USB ports and an ethernet port. Its original purpose was to make USB disks available on the network with CIFS. I reprogrammed it such that it allows GSN to access USB based sensors over the network and when combined with a USB bluetooth adapter it can be used as sink node for a BTnodes network and forward the data over the network to GSN.

Project website: http://gsn.x-way.org/

Source code: http://gsn.x-way.org/

**Compiler construction** / October 2005 – February 2006

Project of the Compiler construction course which had to be done in groups of two students. The goal was to build a compiler from scratch (only relying on some hints and basic class skeletons from the teaching assistants). The compiler is composed of a Scanner, a Parser, a Tree, a Typer and a Generator which are implemented in the Scala language (Scala is an object-oriented functional language and fully interoperable with Java). It compiles a simple object-oriented language into an executable for an artificial RISC processor.

Source code: http://cvs.x-way.org/browse/zweic/