

Webdesc Design

Introduction

The Webdesc system provides online virtual personal computers to users and enables ubiquitous personal computing. The system allows a user to access a fully virtualized computer, from the CPU up. This document will describe the design and implementation decisions that were made while creating the Webdesc product.

Webdesc Internals

Webdesc is made of three large components, as seen in Figure 1. These are the frontend, backend, and controller. The frontend includes the user-visible website and the users' virtual machines (VM). The controller is a simple XML-RPC Server. The backend is a combination of virtualization and clustering software. The frontend and the backend have no communication with each other, and all communication between the two happens through the controller. Each of these three pieces were developed independently of each other and each had their own specific design and implementation details to work out.

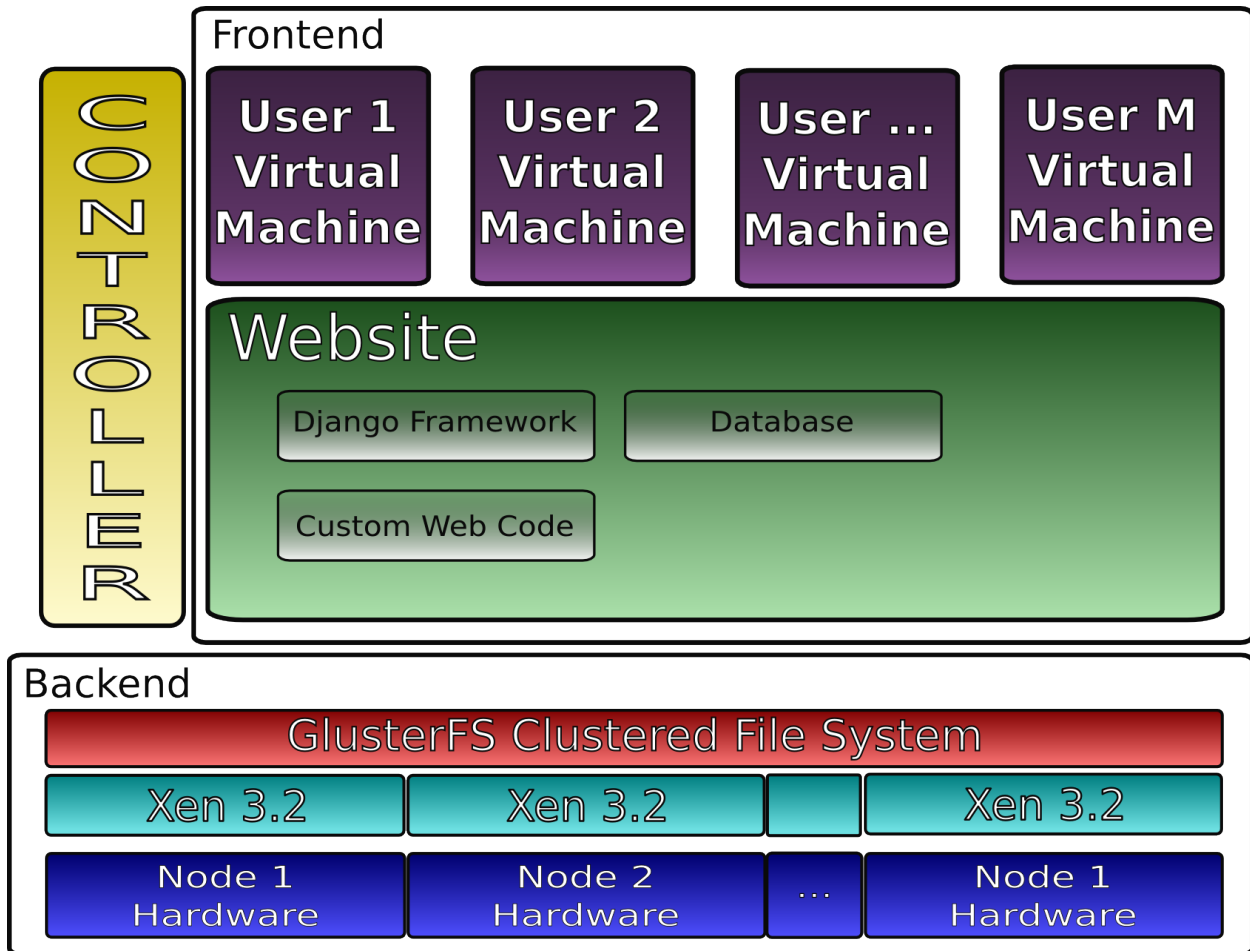


Figure 1

Frontend

The components making up the front end are the user-visible website and the users' VMs. The user's VMs are simply Ubuntu 7.10 (Hardy Heron) virtual machine images. The virtualized nature of the system allows each user to have a virtual machine that they have total control over, even root access! The system is designed to allow any virtual machine image to run, this allows users to have vm's of any operating system, including windows. The website was written using the django framework.[1] The django project is a model-view-controller (MVC) framework. This framework provides object-relational mapping (ORM) using standard python classes and a MySQL database backend. Using django provides user authentication, registration, confirmation, and profiles for free. Django also provides an object oriented model for standard web object like forms, http requests and responses, and urls.

The database schema for the models can be seeing in figure 2.

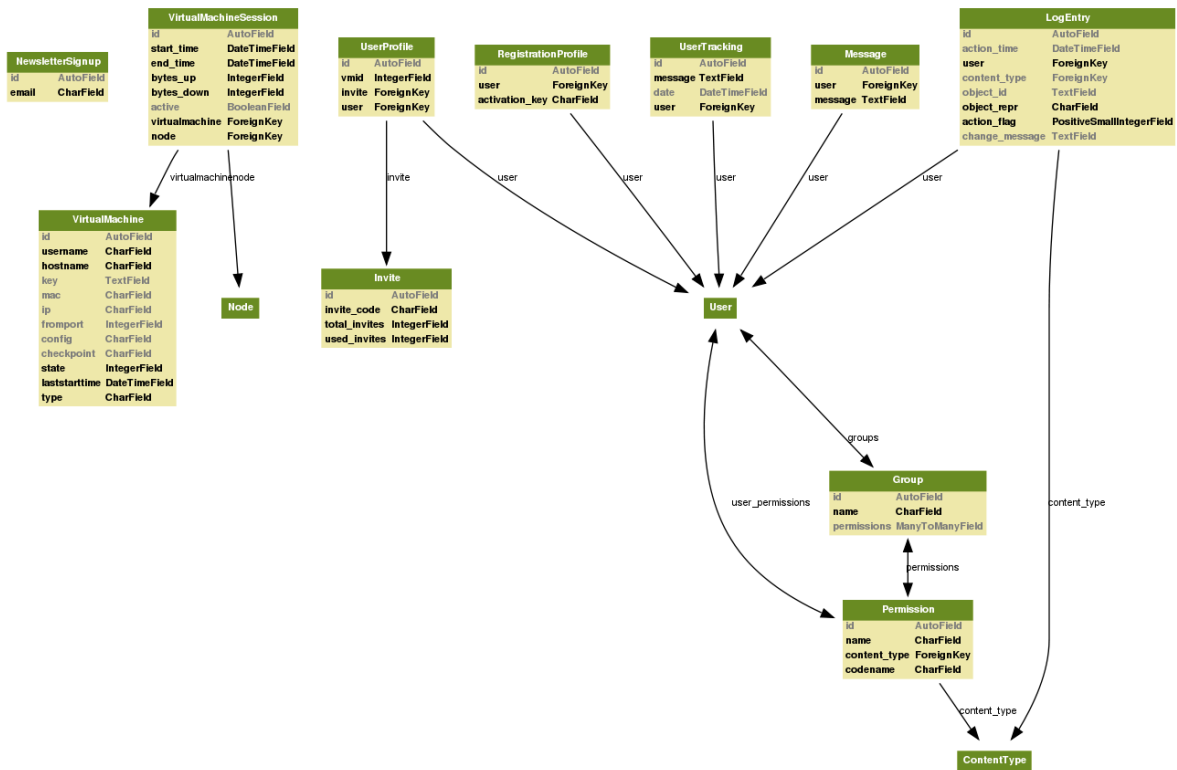


Figure 2

The majority of the website's code is handled by django, or are freely available plugins from the django project's website. Two parts of it were custom coded. The first of these is the main members webpage which allows a user to connect to their virtual machine. This is accomplished by first checking if the virtual machine has been created. If it hasn't, the web code asks the controller to create it. With the virtual machine created the web site then asks the controller to start the virtual machine. The controller sends back information to the website containing information on how to connect to the virtual machine. This information is then passed to a java applet by NoMachine [2] which starts the display connection to the user's virtual machine.

The portion of the code that has been custom coded is the Webdrive. The Webdrive acts as storage for the user which is accessible from both the user's home computer as well as thier Webdesc. This is open-source code which has been adapted to the specifics of our system. This is then also exported by webdav (over HTTPs) which can be mounted both on windows and Linux.

Controller

The controller serves the purpose of managing the virtual machines. The controller performs three main actions. It responds to XML-RPC requests to create and start virtual machines. It also runs a background thread to check if the machine is still being used. If it detects that the connection to the machine has been closed; if it has been disconnected the controller will shut down the corresponding virtual machine.

The XML-RPC server creates and starts virtual machines by talking to the backend. This is currently implemented using remote ssh commands. It first chooses a node of the cluster that is the least loaded, and then runs the command (either to create or to start) the virtual machine on that cluster node.

Backend

The backend of our system is running on a Xen hypervisor[3]. Xen provides the ability to run virtual machines on near bare metal (with especially good performance on specifically tuned Linux kernels). On top of this is a vanilla install of Ubuntu 8.04 LTS (Gutsy Gibbon) which is running Linux kernel version 2.6.24. To provide clustering we have installed glusterfs[4], a clustered filesystem. On the filesystem we store each of the user's virtual machine, as well as a virtual machine for running the website and the controller.

[1] The django projects main website is <http://www.djangoproject.com/>. For a good introduction to django read their overview at <http://www.djangoproject.com/documentation/overview/> .

[2] NoMachine is the software which provides the display connection, you can check out their products at this website: <http://www.nomachine.com/> .

[3] <http://xensource.org>

[4] <http://www.gluster.org/docs/index.php/GlusterFS>