September 30, 2010
Software Architecture Document
Section 6: Views, Revision 1
Deathverse

Baca, Belton, Fishman, Jenne

6. Views

Of the several different views we can describe about Deathverse, we have chosen to detail the following:

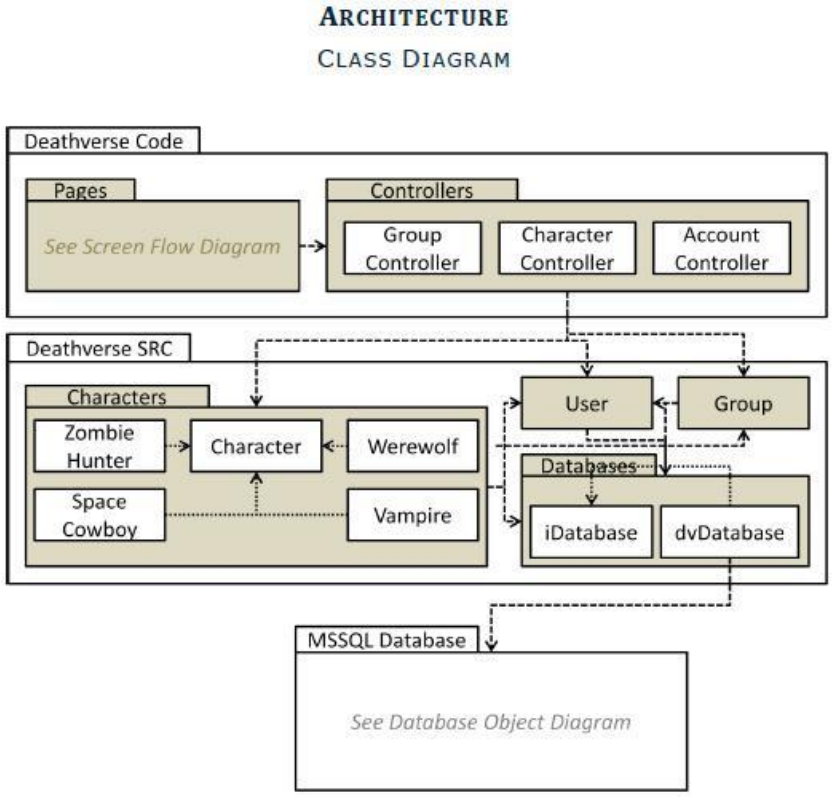
- Module view: This view will give a detailed summary of the static design of the system.
- Component and Connector: This view describes the various interactions between each component of the system, and will give the user a good understanding of where their data flows when they interact with the system.
- Allocation View: This view reveals the various hardware and software requirements that the final structure must meet in order for the system to run as intended.

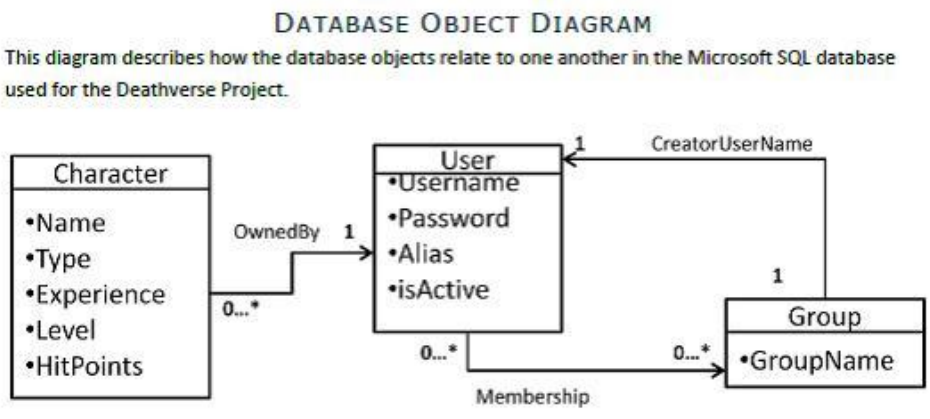
These three views are detailed in the following sections.

6.A Module View

The following diagrams represent design time entities within our system. The UML class diagram and the Database Object diagram provide a static design framework for our system. The class diagram is structured in a manner to show how individual classes are grouped into modules within the software layers of our system.

Module View Primary Presentation





Module View Element Catalog

Elements

This module view expresses the decomposition of the elements in our system into various modules and layers.

The Deathverse Code layer contains the view controllers and the elements of the system that the user actually interfaces with. The Pages module contains the code for the web-pages that users can view while using our system. The Controllers module contains the elements of the top layer that communicate with the middle layer or Deathverse source code layer.

The Deathverse Source layer contains the main functionality of the system. Here the Characters module contains the classes for all of the character objects in the system. The User and Group classes in this layer control management operations for a user's account within the system. The last module in this layer, Databases, controls how the source layer communicates with the underlying database within our system.

Contained in the Database layer are three basic tables that manage user account information and characters contained in the system. All of the operations in the system rely on the data contained in these tables.

Relations

The Deathverse Code layer communicates with the Deathverse Source layer through the Controllers module. The classes in the Controllers module have downward links to the Characters module as well as the User and Group classes in the Source layer. Information requests can only propogate downward through the layers.

Within the Deathverse Source layer, the various character objects in the Characters module inherit from a generic Character class. Most of the classes in this layer are coupled to each other as they require information contained in the various objects to perform the system operations. All of the classes in the Characters module as well as the User and Group classes communicate with the Databases module in order to get information from the actual Database. This module contains the classes the manage the connection and communication between the Source and Database layers.

Within the database exist three tables, one for Character information, one for User information, and one for Group information. A record in the Character table is linked to a record in the User table via an OwnedBy relationship. Furthermore, a record in the User table is linked to a record in the Group table by a Membership relationship.

Interfaces

The only interfaces that exist in our system are the Client server interface, and the application database interface.

Exceptions that can occur within the interfaces include being unable to access the server from the

client side and being unable to connect to the database on the server side. In either case a message is presented to the user of the failed interface connection and no data is altered within the system unintentionally.

With this system being a web-based application, the Client-Server-Database flow is used because it is common and easy practice to develop Internet applications this way. It is beneficial because it effectively separates the interactive user interface, system logic, and data layers making the architecture portable to many different types of applications.

Module View Context Diagram

The UML Class and Database Object diagrams function as context diagrams that show how the modules in our system relate to the environment of the system. The layers of the system all sit on a virtual machine that contains the web-server running the website as well as the database server containing all of the data for the system. All three of the layers in our system sit on this web-server and interact as previously described in the element catalog.

Module View Variability Guide

Within our system, there are few points of variation since all of the layers currently communicate with each other in a linear fashion through direct links. This leads to direct coupling between user operations, module functionality, and database operations.

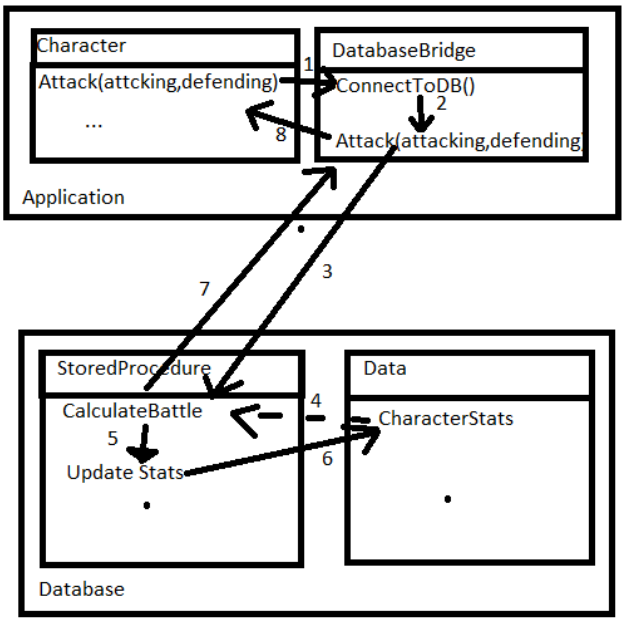
Module View Architecture Background

The design reflected in this view came to be because implementing the system as a model-view controller architecture proved a viable structure for the intended purpose of the project. Deciding on this general architectural strategy allowed us to establish the three layers and then to begin designing the modules contained within the layers and how they would communicate with each other. At that point it was relatively straightforward to design the structure of the database as well as the web-pages which the user would interface with. The design of middle layer of the system which governs the procedures within the system was then driven by the need to connect the front end user interfaces with the back end database. The design of this layer reflects that need to establish proper communications and operations between the layers.

6.B Component-and-connector

The Component and Connector view shows an overhead view of how all the pieces of the system communicate with each other. It reveals the detailed interactions that convert user input into actual changes in the working environment.

C&C Primary Presentation



C&C Element Catalog

Elements

- Application : This is the application which the user works with directly through the web interfaces.
- Database : All of the logic and data storage takes place here
- Character : This is the character object that prompts an attack
- Database Bridge : This is the class that will connect and communicate with the database
- StoredProcedures : This is where all of the logic is stored
- Data : This is where the results and initial information is contained

Relations

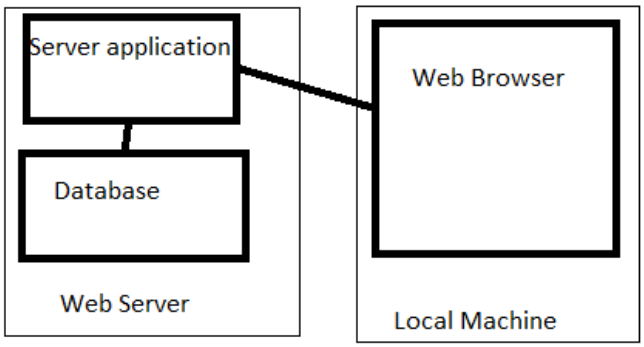
- Characters attack by calling the Database bridge attack command.
- The database bridge attack command will connect to the database and proceed with the attack by calling a stored procedure.
-The stored procedure(SPROC) 'CalculateBattle' will reference the character data from the data component of the DB and then update the data base on the results

-Finally, the calculatebattle SPROC will return and the attackcharacter will redirect the webpage to show the results of the data by referencing the updated data in the DB.

Interfaces

- The only interfaces that exists are the Client server interface, and the application database interface.
- Exceptions include being unable to access the server from the client side and being unable to connect to the database on the server side.
- Because this is a web-based application, this Client, Server, Database flow is used because it very common to develop Internet applications this way. It is beneficial because it effectively separates the Graphic, logic, and data layers which makes the system very portable to other types of applications.

Context diagram

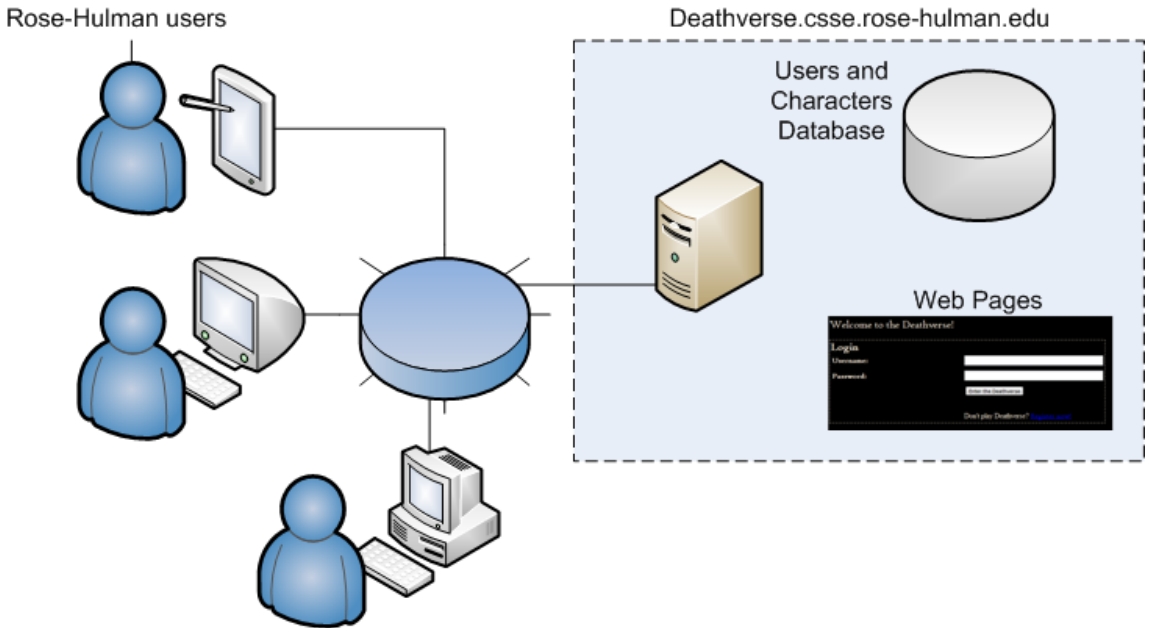


There are 3 elements depicted above. This is the broad view of the system. On the user's local machine they will be running a web browser. The web browser will post messages to the server application on the web server which in turn will access the database and execute whatever commands the user had specified. If we look into the server application we can view one of the main features, attacking an opponent, and all the components and messages passed for this action.

6.C Allocation View

This view details the various hardware requirements for the system to operate. It also describes the various software components that must run on the hardware to ensure the system communicates correctly.

Allocation View Primary Presentation



The above diagram represents Deathverse in its current state. This system follows a client-server model, in which users are able to interact through a web interface. An internal server holds all information for the Deathverse system, including all images, text and website server logic.

Allocation View Element Catalog

Elements

There are two main elements to the Allocation View: the Server, and the Client-side machine.

Server

The server runs Windows XP, and maintains all the information for the Deathverse system. It runs a web server, allowing client machines to connect through HTTP and access the various web pages of Deathverse. It also runs a Microsoft SQL server to hold all character and user information. The current system runs on a Virtual Machine with limited resources, so a modest dedicated machine could run the system in the future if need be.

Client

The Client machine could be one of many different types of machines: PC, Mac, Linux, laptop, desktop, tablet, smartphone... as long as it has a means of reading information from the internet. This requires an internet connection, and a web browser (Internet Explorer, Firefox, Google Chrome, Safari, Opera...).

Relations

The client must send his/her login information to the server, or register on the system first. Once their login credentials have been verified, the client may browse the many pages on the Deathverse system.

Interfaces

There is only two interfaces for this web based system: the client side view (website in a browser), and the database management view.

Exceptions in the Allocation View can occur if there is a disconnect between the client and the server (the network may be down, or simply a cable has been unplugged). The client will be given some sort of error, depending on the machine used to access the webpage.

Allocation View Context Diagram

The diagram at the top of this section serves as the context diagram for this Allocation View. There are two parts: a client machine and a server machine. The client can be running any type of operating system possible, as long as it has a web browser capable of understanding HTTP. The server holds the web server and the database.

Allocation View Variability Guide

The current system is running in a Virtual Machine on a shared computer. So far, the limitations in hardware specs have not caused any problems, but future expansions may benefit from more resources. As long as the server side component is capable of running a Windows operating system with a Microsoft SQL database, future iterations should be compatible with the current version.

Allocation View Architecture Background

For web-based applications, a client/server relationship has been proven to work well in the past. Many large scale systems use this model today, and have been very successful at handling large numbers of users at the same time. We followed this industry standard for our allocation structure to ensure our system could expand if the Deathverse user base increases in the future.

Glossary of Terms

Virtual Machine: a software implementation of a computing platform that acts as a physical machine does.

Web-Server: a computer application that serves web pages when requested by host computers.

Database Server: a computer application that provides the services of a database to other serving applications that request information from it.

Model-View-Controller: a software architecture in which a model is used to manage information, a controller receives input and makes calls to model objects, and a view renders the model into a form that can be interacted with by users.

Microsoft SQL Database: The database server application that is being run on the virtual machine.

Web browser: Uses HTTP to communicate with a server from a client machine

Client: A user's computer that they use

Server: A computer in a remote location

HTTP: Hypertext transfer protocol; it is the standard for communicating over the internet with web sites

Stored Procedure (SPROC): Series of databae query statements that are saved for use at anytime.

Microsoft XP: An Operating System released in August 2001 by Microsoft. It quickly became one of the most popular operating system used in the United States.

Smartphone: a cellular phone with enhanced abilities. It allows the user to do many things that before were only capable on a desktop computer, such as browse the web, and send email, pictures and video.