BUSINESS REQUIREMENTS DOCUMENT

GRID SOFTWARE MANAGEMENT SYSTEM

FOR

ACME CORPORATION

John Stritzinger

CS589

Feb 2nd, 2025

Contents

[STAKEHOLDERS 4](#_Toc189430489)

[CHANGE CONTROL 4](#_Toc189430490)

[SECTION1 - EXECUTIVE OVERVIEW 5](#_Toc189430491)

[1.0 SOFTWARE MODELING 5](#_Toc189430492)

[1.0A -> Executive A. Hypothetical US Core 6](#_Toc189430493)

[1.0B -> Executive B. Modeled Network For This Project 6](#_Toc189430494)

[1.1 PROBABLE HIGH LEVEL NETWORK ARCHITECTURE 6](#_Toc189430495)

[1.2 PROBABLE NETWORK INFRASTRUCTURE 7](#_Toc189430496)

[1.3 AMAZON EC2, AND GOOGLE SPECIFICS 7](#_Toc189430497)

[1.4 ACME’S PREFERENCE 7](#_Toc189430498)

[1.5 OVERALL 7](#_Toc189430499)

[2.0 SOFTWARE REQUIREMENTS 8](#_Toc189430500)

[2.1 Minimum Grid Operations 8](#_Toc189430501)

[2.2 Grid Controllers – Phase I 8](#_Toc189430502)

[2.3 Grid Down State 8](#_Toc189430503)

[2.4 Recovery of Remotes 8](#_Toc189430504)

[2.5 Node Architecture 8](#_Toc189430505)

[2.6 OTHER FEATURES 9](#_Toc189430506)

[3.0 SOFTWARE MANAGEMENT 10](#_Toc189430507)

[3.1 What is an Application? 10](#_Toc189430508)

[3.2 Direct to Database Applications Thick Client to DBMS 10](#_Toc189430509)

[3.3 Direct to Database Applications Via a Code Behind File/Or Server Side Platforms. 10](#_Toc189430510)

[3.4 Web Application to an API via REST 10](#_Toc189430511)

[3.5 Web Application to a PROPRIETARY API INTERFACE 10](#_Toc189430512)

[3.6 Phase I Management – Simplex Applications with Code Behinds. 10](#_Toc189430513)

[3.7 Phase II Management – HA Support with DR Live 10](#_Toc189430514)

[3.8 Phase III – Consideration for Tuxedo Support 10](#_Toc189430515)

[3.9 Example 10](#_Toc189430516)

[3.10 FusionShell 11](#_Toc189430517)

[3.11 cPanel is A Network Management Architecture for Linux 11](#_Toc189430518)

[SECTION4 - Functional Design Patterns 12](#_Toc189430519)

[4.1 BUILDING APPLICATIONS 12](#_Toc189430520)

[4.2 Modeled Networks Assumed for Every Application Installation 12](#_Toc189430521)

[4.3 SAN Provisioning 12](#_Toc189430522)

[4.4 FusionShell Inventory and ITSM 12](#_Toc189430523)

[4.5 SNMP Management 12](#_Toc189430524)

[4.6 All Linux Nodes 12](#_Toc189430525)

[4.7 Software Interfaces 12](#_Toc189430526)

[SECTION5 - DBMS TARGET 13](#_Toc189430527)

[5.0 Minimal Tables Considered 13](#_Toc189430528)

[5.1 Companies 13](#_Toc189430529)

[5.2 Company Billing & Consumer Billing 13](#_Toc189430530)

[SECTION 6 - TEST ENVIRONMENT FOR GRID SOFTWARE MANAGEMENT 13](#_Toc189430531)

[6.1 CAMPUS NODES 13](#_Toc189430532)

[6.2 CTS NODES 13](#_Toc189430533)

[6.3 SD-WAN Connectivity 13](#_Toc189430534)

# STAKEHOLDERS

* Dr. Patterson
* Dr. Vidal
* Portia Plante

# CHANGE CONTROL

* Version 1.0 – February 2nd, 2025

# SECTION1 - EXECUTIVE OVERVIEW

Acme Corporation is our target customer. Acme has a global infrastructure with hosting regions some of which are on-net using their own resources, but Internationally they purchase services from major cloud providers. Furthermore, Acme Corporation has need to connect to ILEC hosting centers where they have specific functions like VPN Management, Remote Access, and Identity Services using Private IP, and in some cases public IP services.

Acme has a US Based Network with a pentagon shape, but one of the pentagon sides is a Disaster Recovery Center, making just four useable targets and one production standby.

Our Goal is to manage software across a hybrid global network in a seamless manner using a Grid Controller Architecture which can run on Virtual Machines or Containers as necessary.

Acme is not unique, they believe their new modern hosting architecture is in-line with other global entities which they have discussed their plans in a working group. They usually do not like deploying new architectures without peer discussions, and consulting engagements with premium vendors each of which are asked to provide a design for review.

Acme is hoping to consolidate several potential designs in a wholistic model which has the accumulated knowledge of its vendors, peers, and internal resources.

Acme has more than 50K employees, more than 20K contractors, and more than 10K B2B Partners which use its portals. Some Elite B2B customers are so important that they need to connect to ACME’s boundary network directly and not via VPN. In this case, DMZ hosts need to be built on the edge of the network at the B2B Customer demarc and not in the Acme Network.

## 1.0 SOFTWARE MODELING

This project assumes that the network will be in place, but wont be used in production until Phase II, and III of this project.

## 1.0A -> Executive A. Hypothetical US Core

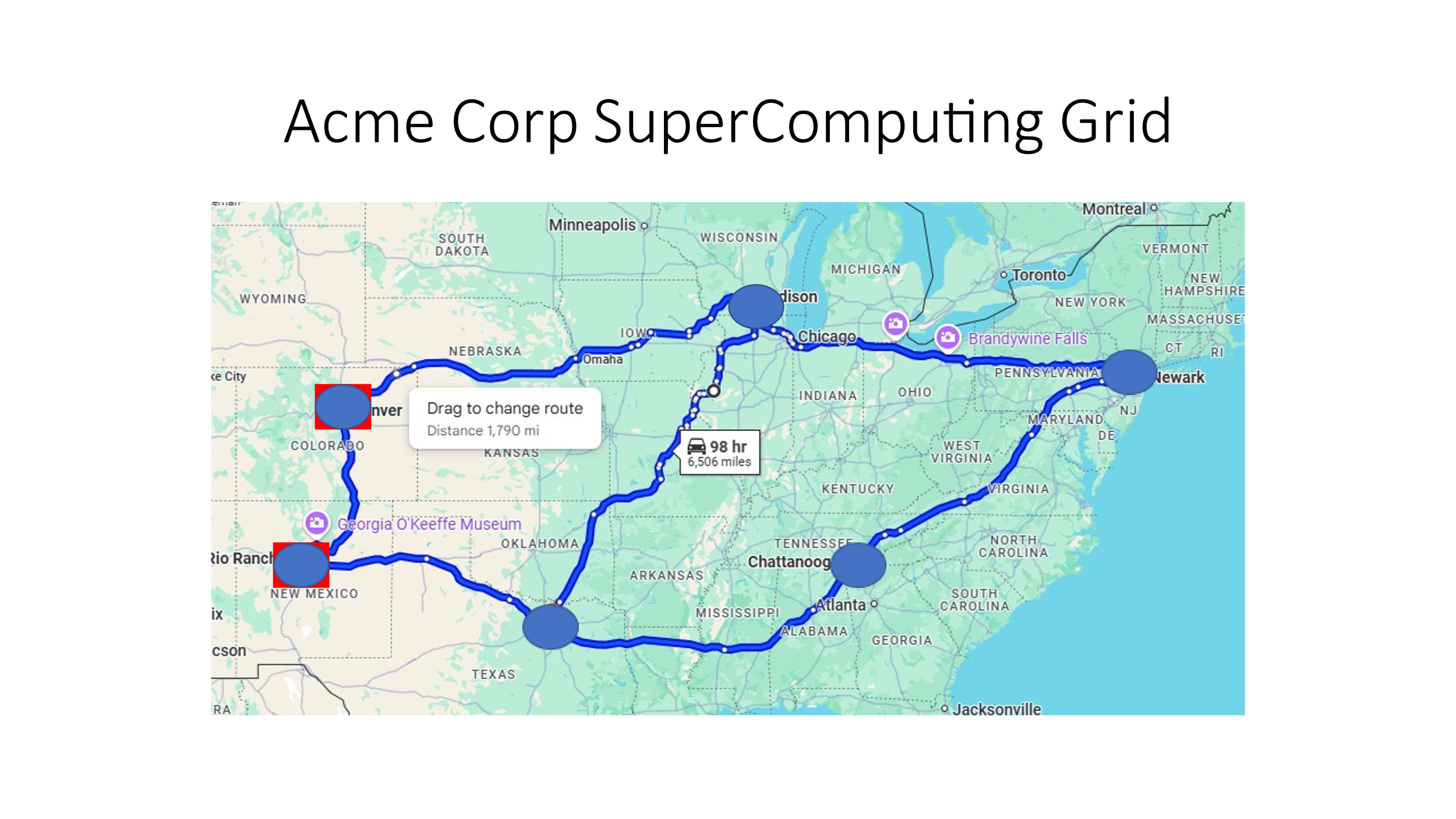


Figure 1.1 – Acme Core Network

## 1.0B -> Executive B. Modeled Network For This Project

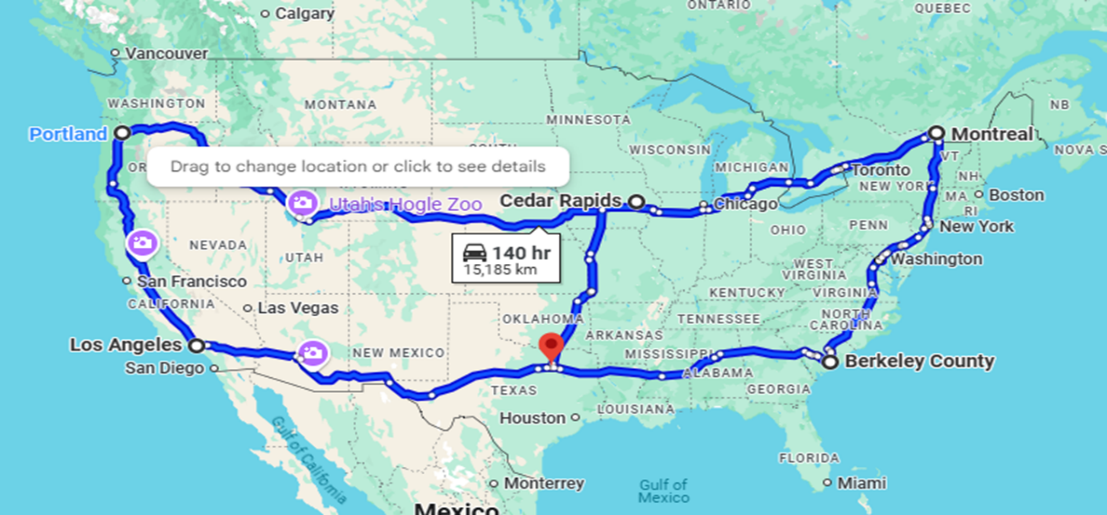


Figure 1.2 – Modeled Network for Design Patterns(Software Engineering)

## 1.1 PROBABLE HIGH LEVEL NETWORK ARCHITECTURE

Since Acme Corporation is a US Flagged company and they own their own optical backbone we are making that a de-facto requirement, noting they made decide to decommission it in the future, but it is not likely because it carries their group bridge calls which includes Classified, and Trade Secret Conversations. Acme has a group connectivity with Global Calls in Excess of 10K people which are scheduled by Division. In other words 5 x 10K group calls must be processed during the same day or simultaneously by different division executives.

1. Assumptions Acme has a US optical core spanning NFL Cities.
2. Acme has primary datacenters in Newark NJ, Madison WI, Chatanooga Tennessee, Rio Rancho New Mexico, and a DR Center in Denver Colorado. We however need to simulate this network with our grid model to prove to them it will work, before they let us install on-premise.
3. Overseas Acme has Active-Active-Active Triangles in Europe, Asia, and South America with one DR in each Region which is a days driving time from each of the others. In areas where the drive is impractical, Cisco or NCR service depots are required.

## 1.2 PROBABLE NETWORK INFRASTRUCTURE

The probable network Infrastructure consists of IBM, HPE, and Cisco Hosts connected over a Cisco and Juniper MPLS Core (P-PE) domestically, Amazon Cloud in Europe, and Google Cloud in Asia.

Acme has the capability of providing Virtual Machines in the form of up to 64GB Memory, Single OS Hosts by up to 64 Cores each. Furthermore, Acme can provision a whole IBM Z16 platform in a single LPAR spanning up to 10K cores or with up to 82 LPARs running a DBMS instance. A Z16 Host Architecture can support 82 DBMS Instances, or 1 with nearly unlimited resources with extension shelves.

## 1.3 AMAZON EC2, AND GOOGLE SPECIFICS

All International Hosts support Public IP addresses terminating to a Virtual Machine. The Operating systems should support IIS web hosts(The Target) running DotNet8, and ASP.Net(4.8+). Acme wants to run C# and VB code behinds off .ASPX pages which can run server side code extensions, without passing connection strings to the individual files. Code Behind files are identified in the header of the .ASPX page and are abstracted from users at runtime. Therefore unlike other Web technologies code-behind server side technology in general cannot be hacked unless the webserver management system is hacked or the file system has invalid permissions.

## 1.4 ACME’S PREFERENCE

Acme would like to build their default apps on Postgres DBMS in grid format, however they will need to be able to install COTS services on Oracle, and MS-SQL Services over time as a 2nd priority.

## 1.5 OVERALL

Grid DBMS instances have to run Active-Active x 4 nodes, with 1 DR site which is managed via automation in some form. The DR architecture can be quite complex, however the DBMS instances should be very straightforward.

# 2.0 SOFTWARE REQUIREMENTS

## 2.1 Minimum Grid Operations

A Minimum Grid will have a controller on each node. In other words the Grid Manager is the first app which has to be present for other apps to be installed. The Grid Manager should run out of band however of the other grid applications (on a different disk, and control plane), and should have the capability to rebuild itself, and the other nodes on a failure. If the rest of the Grid breaks, then the Primary Node needs to run every application in Simplex mode while the rest of the grid is synched back into production. A Primary Node is roughly equivalent to a CPanel/Siteground hosting account and this is a well known process.

## 2.2 Grid Controllers – Phase I

Grid Controllers have to monitor the DBMS across 4 primary, and 1 Dr Nodes. There should be an alert mechanism via SNMP to an HPOV Mirrored Architecture on Failure. HPOV has the ability to escalate unacknowledged alarms to a pager or email process on an interval or rules basis. Grid Controllers should keep track of packet loss, latency, and jitter across the core network to which they are connected. Grid Controller Status Messages should be visible in webmin via a new adapter which should be identical across the architecture.

## 2.3 Grid Down State

The Grid should be deemed down if the DR, and the Primary Host Account are down. This is considered a catastrophic failure in a network with more than 100 applications. The Primary Host therefore has to have hardened operational fabrics.

## 2.4 Recovery of Remotes

The three/four additional Nodes on the Grid Need to be able to be resynchronized in a reasonable period of time by simply backing up existing SQL tables first, and then resynching the primary nodes across the network. Grid Controllers should have these functions built in. IE they should have a Recovery Function which first writes all the current SQL tables to disk like PHPPGADMIN has in its capabilities. Root DBMS files should either be small, or will be installed on SAN disk architectures, which should be able to remirror the partitions within a few minutes to seconds.

## 2.5 Node Architecture

We are defining a Node as a Hosting Account which has awareness of its DBMS, and its other grid nodes, and its controller. This can be achieved by writing custom DotNet code, or by writing an adapter for cPanel to have additional functions in PaperLantern the cPanel Management Tool. Note however that cPanel only supports RedHat variants including Oracle Linux, and Rocky Linux/Centos7-8(EOL). To support other Linux Variants and Windows we are planning to build our own hosting architecture on Windows Server. (This is a nontrivial task). cPanel in comparison has been in development for 20+ years, and has more than 1M customers testing its loads every month. Siteground has migrated from cPanel to their own technology which has been stable for Acme for 3+ years, but at small Public Class services only.

cPanel instances must be VPS instances, as you can not install ANY SOFTWARE ON THEM WITHOUT PLUGINS without your own server. In other words the 10.00 Host Gator Accounts do not work with the architecture we have designed, while full Redhat VM’s do in fact work.

## 2.6 OTHER FEATURES

All Siteground and cPanel requirements for DB Management need to be considered in scope and ssl via certbot provisioning.

# 3.0 SOFTWARE MANAGEMENT

## 3.1 What is an Application?

An Application is a collection of software that communicates to a single database, or multiple databases in its most generic form, however the form of the communication of the software will be over some type of supportable network topology on Standard Cisco, Juniper, and Major Cloud routers, and preferably both. An Application which specifically uses some kind of connectivity that doesn’t work on both router platforms usually is not Internet Standards based, and will not be supportable in practice on a Global network. (See Section 1). Furthermore some IBM and HP-VMS Applications, use a Gateway controller and a tunnel protocol, but if they do most if not all support IP encapsulation techniques. SNA over IP, DECNET over IP etc, FibreChannel over IP as just some examples which all pass services without inspection of the Router Plant over a collection of nodes.

## 3.2 Direct to Database Applications Thick Client to DBMS

Applications can speak from a client to a database using “Thick Client Technology”. These applications have a disk store, and a Database. Examples are ASP.NET Grid DBMS objects, PHP, and NodeJS applications which don’t have an API.

## 3.3 Direct to Database Applications Via a Code Behind File/Or Server Side Platforms.

Application speaking to a Code Behind file usually use Standard HTTP transport, and the WebServer proxies requests to a CGI, NODEJS, or ASP.NET interface inside the Cloud Platform.

## 3.4 Web Application to an API via REST

Application to Native REST Interfaces deliver Data via JSON objects which have to be parsed by the client. Rest Clients can support a variety of backend architectures which manage database insertion.

## 3.5 Web Application to a PROPRIETARY API INTERFACE

It is possible to build applications via SignalR(Microsoft) or other Standards based process and to proxy applications via web or command line tools to a function which responds with an answer. This also includes AI engines which might have custom interfaces and feature calls.

## 3.6 Phase I Management – Simplex Applications with Code Behinds.

In Phase I we are going to support Mirrored 2 Node DBMS’s with an Active-Active Design with flat file dump. For example a Chicago-Dallas Model will support all of North America.

## 3.7 Phase II Management – HA Support with DR Live

We are going to seek to write to a DR node which will be in a unique one-way state. It will accept inserts, but not deletions. In other words, the DR node should have all records on the network in original and modified forms. (Change Control).

## 3.8 Phase III – Consideration for Tuxedo Support

Tuxedo is a requirement for most financial platforms which run at scale which need to batch and reverse 5-1000 records in bulk. This can be done via Tuxedo extensions to standard SQL. (Oracle, MSSQL).

## 3.9 Example

Certification Application For A Corporation, Which Has Internet Users, B2B Users, and Division Users as being built in CS590 this spring and supports an Azure Blob Storage. The requirement is for this app to run with a React Native or Ionic React Front-End to a DotNet API, the form of the Database is not a requirement, so we could see if we can “GRIDIFY” this application by simply migrating the DotNetAPIS to use a different entity Framework set of drivers, and without impact to the controllers running on the Linux platforms natively.

## 3.10 FusionShell

FusionShell is an OpenSource Menu System built by Greenville Associates which has a fixed directory structure, regardless of whether FusionShell is the Front-End Interface, FusionShell normalizes file systems across the enterprise.

## 3.11 cPanel is A Network Management Architecture for Linux

[www.Cpanel.com](http://www.Cpanel.com) for more information. (It’s a Linux Distribution on a Linux Distribution….in other words it’s a controlled version of the Core OS where cPanel manages every file on the platform) except for direct host applications added by the Network Administrator.

# SECTION4 - Functional Design Patterns

# 4.1 BUILDING APPLICATIONS

* Application Type (Direct, DirectCodeBehind, DotNetAPI, NodeJSAPI, PHP, JAVA)
* Database Type (Postgres, SQLite3, MSSQL, Oracle)
* Database Version (19c, 3, SQL2022, etc)
* Cloud Provider (Internal, CTS, Microsoft, Amazon, Google, Verizon, ATT, NTT, CBT, TATA, BT, CAMPUS)
* DBNodeTypes(Have to Be the Same) Per Grid
* GridInstance
* GridInstanceID
* FusionShell Application Version

When an Application is Built on the Grid, it should be stored in two places. In a local version of SQL Lite so if the core network is cut, you should still have a reasonable approximation of your data even if the clients connect asymmetrically. And in the Postgres DBMS.

## 4.2 Modeled Networks Assumed for Every Application Installation

The Modeled Network Architecture should be available in drop downs and lists for the application for provisioning purposes. Imagine you are using Google Console you should assume all nodes are working and active for software delivery perspectives.

## 4.3 SAN Provisioning

SANS Should be considered to Hitachi SAN Nodes, IBM SAN Nodes or Dell-EMC Nodes. Sans should be provisioned in the same locations as the Supercomputers live in an East-West Topology at a minimum. 2 Nodes per site.

## 4.4 FusionShell Inventory and ITSM

FusionShell Professional ITSM and Inventory will be Assumed for Support the Single View of the Customer.

## 4.5 SNMP Management

HPOV is assumed as the SNMP Manager in an HA Model in the Columbia, SC Operations Zone to start and will be active and working in test.

## 4.6 All Linux Nodes

Will run a Grid DBMS instance, a Controller(which might communicate to the IIS Hosts as well via Agents), and a cpanel Hosting Account with Postgres DBMS Support. Each node will have Webmin installed and available on port 8080 across both private and public infrastructure.

## 4.7 Software Interfaces

Need to support SSO and Social Media Logins to Google, Facebook, and Twitter.

# SECTION5 - DBMS TARGET

## 5.0 Minimal Tables Considered

We are planning on support Postgres SQL. Minimal Tables Shall be:

* Public Console Users
* Administrative Console Users
* Companies
* Business Units
* Business Users
* Grids
* Nodes (NodeID, GridId)
* Locations
* CloudProviders
* Applications
* DBMSInstances
* Hosts
* Clouds
* VPCs
* VMs

## 5.1 Companies

Companies should list users and their ERP Login Types (Oracle, Aloha/NCR, MicrosoftDynamics).

## 5.2 Company Billing & Consumer Billing

Public Console Users not related to a company should have a direct billing mechanism

(Credit Card, Paypal, Stripe)

# SECTION 6 - TEST ENVIRONMENT FOR GRID SOFTWARE MANAGEMENT

## 6.1 CAMPUS NODES

USC Datacenter at Swearingen is hosting 2 VMS, (1 Windows, 1 Ubuntu 22) Version of Linux.

## 6.2 CTS NODES

CTS is hosting 2 VMS one shared one dedicated at Barnwell Colo at 1800 Washington Street & Barnwell.

## 6.3 SD-WAN Connectivity

SD-WAN Connectivity is being provided by SanDiego Based Zerotier Networks via a Software Router Architecture.