Hey Himanshu Chauhan **thanks for your request you need to sign a Document** for this . Your Lot is and Division is Testing Document Test Set 1: Fault Tolerance Test 1: Basic Failover Test Description: This tests that the client can handle a basic process crash fault. In the event of a process crash, the client should switch from primary to backup and continue operating. Test Procedure: 1. Start up all components; however skip starting of the executives. We don't need to start the executives because this test does not test automatic recovery. Script clean up Script runServer1 starts the primary. Script runServer2 starts the backup. Script runclient starts the client. 2. Crash Server1 using kill-9 Expecting to see from client window: "Unable to call TrafficControl_PositionUpdate , Exception:IDL:omg.org/CORBA/COMM_FAILURE:1.0 Oops..Server crashed..Switching to Replica.. Resolving TrafficControl object.... Calling ReplicationManager::GetPrimaryServer... Replication Manager tells me to switch to TrafficControlServer2." Expected to see on sacred machine: "ALARM: server 1 is DOWN primary change from: 1=>2 Client contacted me and GetPrimaryServer returns: 2" 3. Verify: a. The client catches the resulting exception. b. The Replication Manager tells Server2 to become the primary c. Server 2 loads the state from the database. d. The client contacts the Replication Manager and switches over to Server2. Test Set 2: Recovery Test 2.1: Manual Recovery to a Second Machine Test Description: This tests that the client can handle a crash sequence. Test Procedure: 1. Start up all components; however skip starting of the executives. We don't need to start the executives because this test does not test automatic recovery. Script runServer1 starts the primary. Script runServer2 starts the backup. 2. Crash Server1 using kill-9 3. Verify: a. The client catches the resulting exception. b. The Replication Manager tells Server2 to become the primary c. Server 2 loads the state from the database. d. The client contacts the Replication Manager and switches over to Server2. 4. Restart the killed process on a new machine (any one of the Game machines) . Log into the machine and run script runServer1. 5. Verify that the client continues to talk to Server 2. 6. Crash Server2 using kill-9 7. Verify: a. The client catches the resulting exception. b. The Replication Manager tells Server1 to become the primary c. Server 1 loads the state from the database. d. The client contacts the Replication Manager and switches over to Server1. Test Set 3: Exception Handling Test 3.1 Out-of-Order Startup Sequence Test Description: The system should gracefully handle cases where components are started out of order. Test 3.2 Test Death of Naming Service Test Description: The naming service will live on a sacred machine, but the servers should gracefully handle the case where the naming service becomes unavailable. Test Procedure: 1. Start up all components normally. 2. Kill the naming service. 3. Verify a. The Servers catch the exception when trying to contact the naming service. b. The replication manager catches the exception when trying to contact the naming service. Test 3.3: Test Death of Database Test Description: The database server will live on a sacred machine, but the servers should gracefully handle the case where the naming service becomes unavailable. Test Procedure: 1. Start up all components normally. 2. Kill the database server 3. Verify a. The Servers catch any exceptions when trying to contact the database Test 3.4: Test Death of Replication Manager Test Description: The replication manager will live on a sacred machine, but the servers should gracefully handle the case where the naming service becomes unavailable. Test Procedure: 1. Start up all components normally. 2. Kill the replication manager. 3. Kill the primary server. 4. Verify a. The client handles any exceptions when trying to contact the replication manager Out of Scope Things we don't handle: • We don't handle starting multiple servers with the same name. So don't try runServer1 more than once or on multiple machines. • If you kill the database, we do not guarantee correct behavior. Clients should be able to continue to update their local position, but they cannot join appropriately. • We don't necessarily handle the case where both servers become unavailable at once. • We don't handle misbehaving clients. Clients are expected to abide by the protocol defined in the IDL. • We don't handle client crashes. If a client joins the server and then fails, the client will remain in the system.