**DISTRIBUTED VERSION CONTROL SYSTEM**

**ABSTRACT:**

Version control systems are used to maintain source code in a collaborative project. If ten people are working on the same project and each of them works in his/her own part of the project, we should be having a situation where all ten of them are working on their own copy of the code. So it is essential to ensure that the changes made by the other developers are also reflected in their local version of the source code. This is where a Version Control System comes into existence.In case of a distributed version control system, there are a lot of synchronization problems. This project aims at bringing the maximum synchronization among the various users of the code.

**SCENARIO:**

Imagine that there are five users who work on the same project in a distributed environment. Among the five users a ‘central repository’ is chosen using an election algorithm. A central repository maintains the latest updated version of the code. It also maintains different versions of the code. Suppose if a user makes a change in the code, he has to send the changes to the central repository. The central repository does the following:

* It checks whether the code is the latest one, if not it notifies the user that it is not the latest version. It finds the differences and sends it to the user.
* If it is the latest version then it updates its own code and creates a new version.

The central repository gives a version number to the code whenever it is modified. So it can backtrack through the previous versions and find out which version the user has. Whenever a new version is created, it notifies all the users to update its local version.

Consider the following scenario where A, B and C are working on the local versions of the same code:

Developer A: Devleoper B: Developer C:

#include<iostream> #include<iostream> #include<iostream>

#include<string>

using namespace std; using namespace std; using namespace std;

int main() int main() int main()

{ { {

cout<<”hai”; cout<<"hai"; return 0;

return 0; string a=”test”; }

} return 0;

}

Now all the three sends a request to the central repository to update the code at the same time. A request is send in form of a hash. The hash is calculated based on the time of request and version of the code. After the request is received, suppose that A and C uses the latest version, then it arbitrarily chooses anyone and updates it and sends an acknowledgement to the developer. If B does not have a latest version, then the central repository first informs B that it does not have the latest version and sends the latest version. Now B makes changes in the latest version and resends it. Now the code is updated.

**PROBLEMS:**

* Suppose if the central repository fails before notifying the users about the latest version, then each user will be having a different version. So again a central repository is chosen and all the users send their local versions of code to the central repository. A new collaborative version is created and all users are notified.
* Suppose if the user fails when the central repository notifies about the new version, it will have the old version. So if the failed user comes back, it must notify about the latest version.

**ALGORITHMS USED:**

Election algorithm