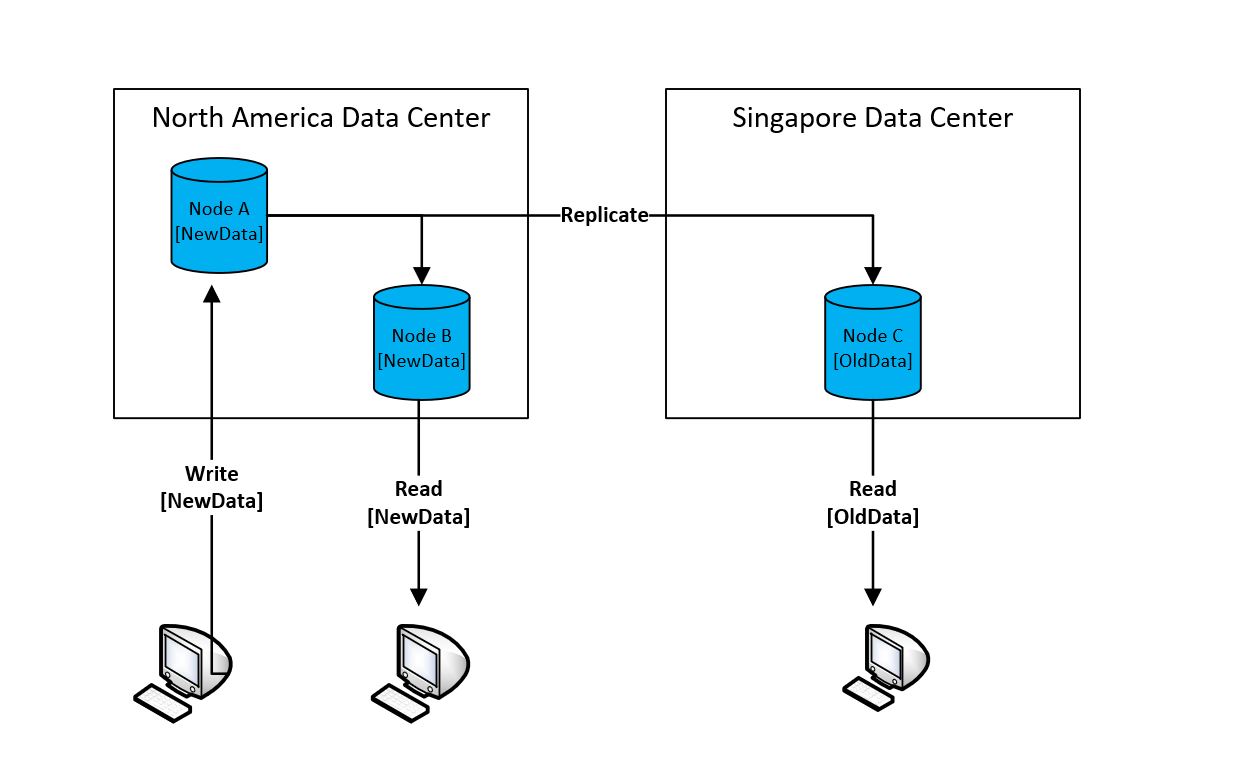
**Eventual Consistency(Nihai Tutarlılık)**

Herhangi bir zamanda sistemi tamamen güncellemeye kapatsanız bile herhangi bir veri yeni bir değeriyle güncellenmese bile sistem eninde sonunda mutlaka doğru değere ulaşacağını bu tutarlılık modeli bize gösterir.Veri kaybının hiçbir şekilde olmayacağının garantisinide bizlere veriyor.Peki Nihai Tutarlılık (Eventual Consistency) nerelerde geçerlidir?.Verinin anlık(real-time) doğruluğunun büyük önem arz etmediği sistemlerde geçerlidir.Amaç anlık tutarlılık değil sonuçda tutarlılığı elde etmektir.Yani Nihai tutarlılık(Eventual Consistency), dağıtılmış bilgi işleminde, belirli bir veri öğesinde yeni güncelleme yapılmazsa, o öğeye tüm erişimlerin en son güncellenen değeri döndüreceğini gayri resmi olarak garanti eden yüksek kullanılabilirlik sağlamak için kullanılan bir tutarlılık modelidir. Tutarlılık modelleri genellikle geleneksel ATİD (Atomiklik, Tutarlılık, İzolasyon, Dayanıklılık) garantilerini verir ama Nihai Tutarlılık aksine TYS (Temelde Kullanılabilir, Yumuşak durum, Sonunda tutarlılık) garantisini bizlere verir.İnsanlar neden Nihai tutarlılığı isterler?.Siz bir versiyon kontrol sisteminde çalışıyorsanız kendi bilgisayarınıza arkadaşınızın çalıştığı bilgisayarın bir kopyasını alırsınız.Bu arkadaşınızın kopyasında tamamen bağlantı kopuk bir şekilde çalışırsınız ve daha sonra onla birleştirme işlemi gerçekleştirirsiniz.Bu birleştirme işlemini gerçekleştirirken en son aşamada bunun tutarlılığının doğru çalışmasını istersiniz.Siz bilgisayarınızda çalışırken sürekli başkalarının yaptığı güncellemelerin sizi etkilemesini istemezsiniz.Yani bağlantısız bir operasyonunuz varsa Nihai Tutarlılık faydalı bir tutarlılık olarak karşımıza gelir.İkinci önemli nokta ise Nihai Tutarlılıkta hızlı okuma ve yazma işlemini gerçekleştirmemiz.Peki bu tutarlılık modelinin hiç mi olumsuz yönü yoktur?.Nihai tutarlılık bazen dağıtılmış yazılım uygulamalarının karmaşıklığını arttırdığı için eleştiriliyor.Bunun nedeni kısmen tutarlılığın olması ,güvenlik garantisi vermemesi.Aşağıdaki şekil bize Nihai Tutarlılıkla çoğaltma kavramını göstermektedir.Çoğaltmalar her zaman okunabilir olsada bazı çoğaltmalar belirli bir zamanda kaynak düğümdeki en son yazmayla tutarsız olabilir.Diyagramda Düğüm A kaynak düğümdür ve düğüm B ve C kopyalarıdır.

****

**Çatışma çözümü(Conflict resolution)**

Dağıtılmış sistemlerde uyuşmazlık çözümünü ele almak istiyorum. Yani, farklı insanlar verileri bağımsız olarak değiştirirse ve meydana gelen çatışmaları nasıl çözeriz?.Çoğaltma yakınsamasını sağlamak için bir sistemin dağıtılmış verilerin birden çok kopyası arasındaki farkları uzlaştırması gerekir ve bu iki bölümden oluşur.

\*Sunucular arasında verilerin sürümleri veya güncellemelerini değiştirmek

\*Eş zamanlı güncellemeler gerçekleştiğinde,mutabakat(reconciliation) adı verilen uygun bir son durumun seçilmesi

Mutabakat(reconciliation) için en uygun yaklaşım uygulamaya bağlıdır.Yaygın bir yaklaşımın biri “son yazar kazanır” yaklaşımı.Diğeri ise kullanıcı tanımlı bir çakışma işleyicisini çağırmaktır.Zaman damgaları(timestamp) ve vektör saatler(vector clocks) genellikle güncellemeler arasındaki eşzamanlılığı tespit etmek için kullanılır.Bazı insanlar “son yazar kazanır” yaklaşımını kabul edilemez bularak “ilk yazar kazanır” yaklaşımını benimser.Eşzamanlı(concurrent) yazmaların mutabakatı bir sonraki okumadan bir süre önce yapılmalıdır ve farklı örneklerde programlanabilir.

**Güçlü Nihai Tutarlılık (Strong Eventual Consistency)**

Nihai tutarlılık yalnızca bir canlılık garantisidir.Güçlü Nihai Tutarlılık(SEC) aynı güncelleştirme kümesini alan her iki düğümün aynı durumda olacağına dair güvenlik garantisini verir.Ayrıca,sistem monoton ise,uygulama hiçbir zaman geri dönüşlere maruz kalmaz.Çakışmasız çoğaltılmış veri türleri Güçlü Nihai Tutarlılık’ın sağlanmasında yaygın bir yaklaşımdır.

**DROPBOX**

Dropbox, çoğu durumda senkronizasyon yoluyla Anında Tutarlılık(immediate consistency) sağlamış gibi gözüksede Nihai Tutarlılık(Eventual Consistency) ile çalışıyor.Cihazlarınız arasında (pc, tabletler, akıllı telefonlar) bağlantı olmadan bile kullanabileceğiniz dağıtılmış bir sistemdir.Dropbox'ın yerel klasöründeki dosyalar üzerinde işe giderken, galeride, denizde, 3G'nin kullanılamadığı dağlarda veya daha sonra kullanmak üzere pil veya bant genişliği kaydetmek istediğinizde çalışabilirsiniz.Böylece Dropbox size senkronize edilecek bir klasör vermeyi vaat eder.Delta sıkıştırma algoritmaları ve çakışmaların yönetimi veya mümkün olduğunda LAN üzerinde senkronize edilmesi ve başka bir durumda İnternet üzerinden senkronize edilmesi gibi birçok karmaşıklığı gizler.Dropbox Nihai Tutarlılığı kullanır.Ctrl + S tuşlarına her bastığınızda ve güncellemeler diğer cihazlara aktarıldığında kelime işlemciniz donacağı için İnternet gibi büyük bir ağ üzerinden işlem tutarlılığı elde etmek mümkün olmaz. Ağda, çalışan bir bağlantının olmaması gibi bir bölünme olması durumunda, anında tutarlılık elde etmek de imkansızdır.Sonuç olarak ne yapılır? Bağlantı mevcut olduğunda güncellenecek olan belgeniz üzerinde çalışırsınız. Ve sonra bağlantı mevcut olduğunda diğer cihazlarınızda güncellenir. Yüklenecek büyük yeni dosyalarınız varsa bazen senkronize etmek çok zaman alır. Ancak aynı anda birden fazla cihazdan olmasa da çalışmaya devam edebilirsiniz.Dropbox böylece Anında Tutarlılıktan açıkça kaçınmaya ve ondan kurtulmaya dayanan Nihai Tutarlılık modelini kullanır.Sonuç olarak Nihai tutarlılık, belirli bir veri öğesinde yeni güncelleme yapılmazsa, o öğeye tüm erişimlerin en son güncellenen değeri döndüreceğini gayri resmi olarak garanti eden yüksek kullanılabilirlik sağlamak için dağıtılmış bilgi işlemde kullanılan bir tutarlılık modeli olduğunu anlamış oluyoruz.