



Windows Azure ile Cloud Computing Uygulamaları

10 Agustos 2013(v0.1)

İbrahim ATAY



Windows Azure ile Cloud Computing Uygulamaları

© 2013 İbrahim ATAY

Bu kitabın tüm yayın hakları İbrahim ATAY'a aittir. Kendisinden izin alınmadan bu eserden kısmen veya tamamen alıntı yapılamaz, hiçbir şekilde kopya edilemez, çoğaltılamaz ve yayınlanamaz.

Birinci Yayın: 2013 / Türkiye

ISBN: 978-605-64256-0-8

Windows Azure ile Cloud Computing Uygulamaları - 1

Cloud Computing mimarisi, 1970 yılından başlayarak gelişen süreç içerisinde kullanılan alt yapı sistemlerin geliştirilmesi ve eksiklerinin giderilmesi ile ortaya çıkarılmıştır. Bu yazılım ile Cloud Computing mimarisi ve Windows Azure Platform ile ilgili anlatımlar gerçekleştirilmiştir.

Windows Azure ile Cloud Computing Uygulamaları - 2

Cloud Computing mimarisi, geliştirici yönünde birçok problemi çözümlenmektedir. Söz konusu çözümler ile birlikte terminolojimize yeni kavramlar ve yaklaşımlar eklenmiş bulunmaktadır. Bu yazımda Cloud Computing sistemleri üzerinde bulunan uygulamaların Scalability süreçlerine değinmiş bulunmaktayım.

Windows Azure ile Cloud Computing Uygulamaları - 3

Şirketler iç ve dış operasyonları yönetmek amacı ile çeşitli enstrümanlar kullanmaktadır. Operasyon süreçlerinin değişmesi, uygulama sistemlerinin de değişimlere adapte olabilmesi gerekmektedir. Söz konusu süreç işletmelerin sürekliliğini devam ettirebilmesi için büyük önem taşımaktadır. Windows Azure Platformu, şirketler için birçok kolaylıklar ve minimum maliyet ile şirketlere özgür çalışma imkanı sağlayabilmektedir

Windows Azure ile Cloud Computing Uygulamaları - 4

Uygulama geliştirilmesi ya da mevcut uygulamaların Windows Azure Platform 'a taşınması ile ilgili birçok senaryo üretilebilir. Geliştirilen senaryolar; zaman, maliyet ya da süreçlerin basit olarak gerçekleştirilebilmesi mümkün olmaktadır. Örnek verilen senaryolar, istek ve mevcut koşullara bağlı olarak şekillendirebilecektir. Windows Azure Platform 'u ile şirketler minimum maliyet ile özgür çalışmalarına olanak sağlamaktadır.

Windows Azure ile Cloud Computing Uygulamaları - 5

Mesleki gerekliliklere bağlı olarak şekillenen hayatlarda, adaptasyon öğrenilmesi zor olmasına karşın, çözümleri de yanında getirmektedir. Yeni bir adaptasyon sürecine girmektedir. Ama geçmişte sahip olduğumuz bilgilerin üzerine yeni kat oluşturmaktayız. "WindowsAzure.FunnyApp" projesi ile Windows Azure Platform ve Cloud Computing mimarisini anlayarak. Eğlenceli olarak sürece adapte olmayı amaçlamaktayız.

Windows Azure ile Cloud Computing Uygulamaları - 6

İş uygulamaların başarılı ve performanslı çalışması, uygulama yazılım mimarisi ve konumlandırıldığı platform ile ilişkili olarak gerçekleşmektedir. Windows Azure Cloud Service uygulama geliştirme modeli ile geliştirilen uygulamalar, iş sürecine bağlı olarak parçalarına ayrılarak, beklenen başarı ve performans elde edilebilmektedir.

Windows Azure ile Cloud Computing Uygulamaları - 7

Windows Azure Storage yaklaşımı, geliştiricilerine kullandıkları verilerin sınıflandırma ve planlamaya yöneltmektedir. Veri yığınlarının, Windows Azure Storage uyumluluğunun sağlanması ile esnek ve kolay yönetilebilir yapılara sahip olunmaktadır.

Windows Azure ile Cloud Computing Uygulamaları - 8

İş uygulamaların başarılı ve performanslı olarak çalışabilmesi için geliştiricilerin, iş tecrübelerine göre çeşitli yaklaşımlar uygulanmaktadır. Günümüzde depolama ve verinin kullanımı konusunda yeni konseptler oluşturulmaktadır. Microsoft Windows Azure Platform ile de sürece yeni bir bakış açısı oluşturulmuştur.

Windows Azure ile Cloud Computing Uygulamaları - 9

İş coğrafyaları ve gereksinimleri her gün farklılaşıyor. Süreçlere uyum sağlamak her zamanınkinden zor ve maliyetli olmaktadır. Günümüz şartlarında şirketlerin ekonomik, kaliteli ve sürdürülebilir altyapılar sahip olmasının en kolay yolu Cloud Computing den geçmektedir. Bu bölümde Windows Azure Platform üzerine Windows Azure Cloud Services konsepti ile geliştirilmiş "WindowsAzure.FunnyApp" uygulaması adım adım yayınlama süreçlerini incelenmiştir.

Windows Azure ile Cloud Computing Uygulamaları - 10

Maliyetlerin yüksek olduğu dönemimiz de şirketler, farklı kariyer odaklı kişileri outsourcing ederek çalıştırma şansı bulabilmektedir. İş süreçlerin yönetebilmek için kullanmış oldukları altyapı sistemleri ise, Cloud Computing hizmetleri ile sağlamaya çalışılmaktadır. Bu bölümü ile şirketler için Cloud Computing altyapısının önemini ve Windows Azure Platform ile kazanacakları bazı olanaklar hakkında bilgi verilmiştir.



Windows Azure ile Cloud Computing Uygulamaları – 1

Cloud Computing mimarisi, 1970 yılından başlayarak gelişen süreç içerisinde kullanılan alt yapı sistemlerin geliştirilmesi ve eksiklerinin giderilmesi ile ortaya çıkarılmıştır. Bu yazılım ile Cloud Computing mimarisi ve Windows Azure Platform ile ilgili anlatımlar gerçekleştirilmiştir.



Toplumların büyüme ve değişmesi kaçınılmaz süreçtir. Kıyafet almak için dışarı çıkıp, mağazaları gezdiğimiz günler uzak olmasa gerek. Geçmişte internet de kıyafet satılacağına inanmayan kişiler, günümüzde internet mağazaları ile son modayı takip etmektedirler. Saatlerce aradığımız bir kitabı, kitapçıya gidip kitabı anlatıp, kitapçıda var olup olmadığını anlamak ile uğraşmak yerine artık kendi kitapçımız olduk. Aramış olduğumuz kitabı, birkaç internet sitesi dolaşarak zaman kaybetmeden satın alabiliyoruz.

Geçmişten günümüze birçok süreç ve deneyimler ile büyük mağazalardan / kitapçılardan, büyük internet sitelerine terfi etmiş olduk. Süreçlerin bu şekilde ilerletilmesi ile birlikte, internet mağazalarının tüketim toplumundaki yeri büyümeye devam etmektedir.

Büyümeye devam eden internet mağazaları / kitapçılar / sosyal ağlar sahiplerden, şirketlerden çıkarak internet kullanıcılarına ait alanlar haline geldi.

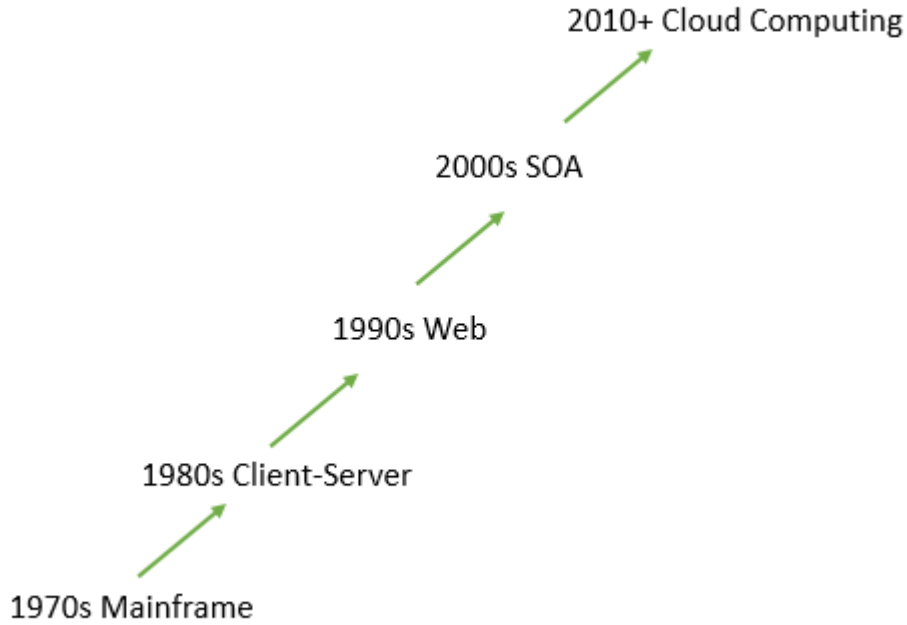
Giderek artan müşteri taleplerine bağlı olarak mağazalar, yeni çalışanlar olarak müşterilerine daha iyi hizmet etmek ve daha çok satış yapabilmek amacı içine girmişlerdir. Söz konusu süreç internet ortamında olan bir mağazayı düşündüğümüzde ise, yeni sunucular ve sunucu yöneticileri işe alınacaktır.

İnternet alanlarının sunucu alma yarışı zaman içerisinde büyük MainFrame mimarilerinin kurulması gerektirdi. İçerisinde bulunduğu süre en iyi çözüm olarak gösterilebilecek mimari olarak söylenebilirdi.



(IBM 7090 - Stanford University)

Artan kullanıcı talepleri, zaman içerisinde uygulama geliştirme süreçlerini de büyük oranda etkileyerek, yeni çözümlerin ortaya çıkmasına yol açmıştır.



Kullanıcı taleplerinin yoğunluğunun artmasına bağlı olarak 1970 ve öncesinde birçok çözüm ile talepler karşılanmaya çalışılmıştır. 1970 yılında mainframe kurulması ile sunucuların birbirine bağımlı olarak çalışma stratejisi, 1980 yılında yerini Client-Server mimarisi ile dağıtık uygulamalara bırakmıştır. Söz konusu değişimler, geliştirilen uygulamaların ve donanım mimarilerinin yeniden şekillendirilmesine neden oldu.

İş ve kişisel uygulamaları istedikleri yerden erişme yetersizliği nedeni ile 1990 yılı ile Web(Internet) uygulamaları gündeme gelmeye başlamıştır. 1970 ve 1980 yıllarında benimsenen uygulama geliştirme yaklaşımları Web(1990s - Internet) devrimi ile yetersiz kalmaya başlanmıştır. Söz konusu süreç ile yeni uygulama geliştirme mimarisi olan SOA mimarisi ortaya çıkmıştır.

1970 ile başlayarak 2010 yılına geline süreç içerisinde birçok geliştirilmiş uygulama ve şirketlerde biriken sunucu alt yapı dağları meydana gelmiştir.

Gelişim sürecinde 2010 yılına gelindiğinde kullanıcıların Web(Internet)'te geçirilen sürenin artması ve iş uygulamalarında yoğunlukların artması Cloud Computing mimarisinin oluşturulmasına neden olmuştur. Cloud Computing mimarisi 1970 yılından başlayarak 2010 yılına kadar olan süre içerisinde geliştirilen mimari yaklaşımları ve donanım alt yapılarını temel alarak geliştirilmiştir.

Cloud Computing 'in temellerini oluřturulan mimariler zaman ierisinde birok deneyim ve alt yapı problemleri ile eřitli konularda hizmetler vermiřtir. Sre ierisinde oluřturulan zmler ynetim, enerji, lisanslama ile ilgili alıřmalar gibi birok maliyet, zaman ve gvenlik gibi konulara problemler ortaya ıkarmıřtır. Cloud Computing mimarisi ise gemiř dnemlerde meydana gelen problemlerin zm olarak kařımıza ıkmaktadır. Konu ile ilgili rnek vermek gerekir ise;

Kıyafet satıřı yapan internet uygulaması ele alındıėında, uygulamaların sunulması ve desteklenmesi amacı ile bazı satın alım iřlemlerinin yapılması gerekmektedir. Yapılması gereken satın alma gereksimleri bulunmaktadır. Satın alma gereksinimleri ile ilgili olarak temel liste ařaėıdaki gibi olmaktadır.

- 10 adet sunucu makine
- 10 adet sunucu kurulum maliyeti
- 10 adet makine zerinde kullanılması amacı ile sunucu iřletim sistemi temini
- Gvenlik yazılımlarının temin edilmesi
- Sunucu ynetim, bakım ve gvenlik iřlemlerini gerekleřtirecek ekiplerinin oluřturulması

Yukarıdaki belirtilen rnek gereksimlerin saėlanması ile kıyafet satıř sreci bařlamıřtır. Sre bařlamıř ve uygulama durum analizi yapıldıėında ařaėıdaki gibi bazı sonular oluřmaktadır.

- Kurulmuř olan alt yapı gereksinimlerin zerindedir.
- Kurulmuř olan alt yapı gereksimleri karřılamamaktadır.
- Kıyafet satıř uygulamasına yeni sunucular gerekmektedir.
- Sunucu ynetim, bakım ve gvenlik maliyetleri srekli olarak artmaktadır.
- Enerji problemleri yařanmaktadır.
- Lisans maliyetleri artmaktadır.

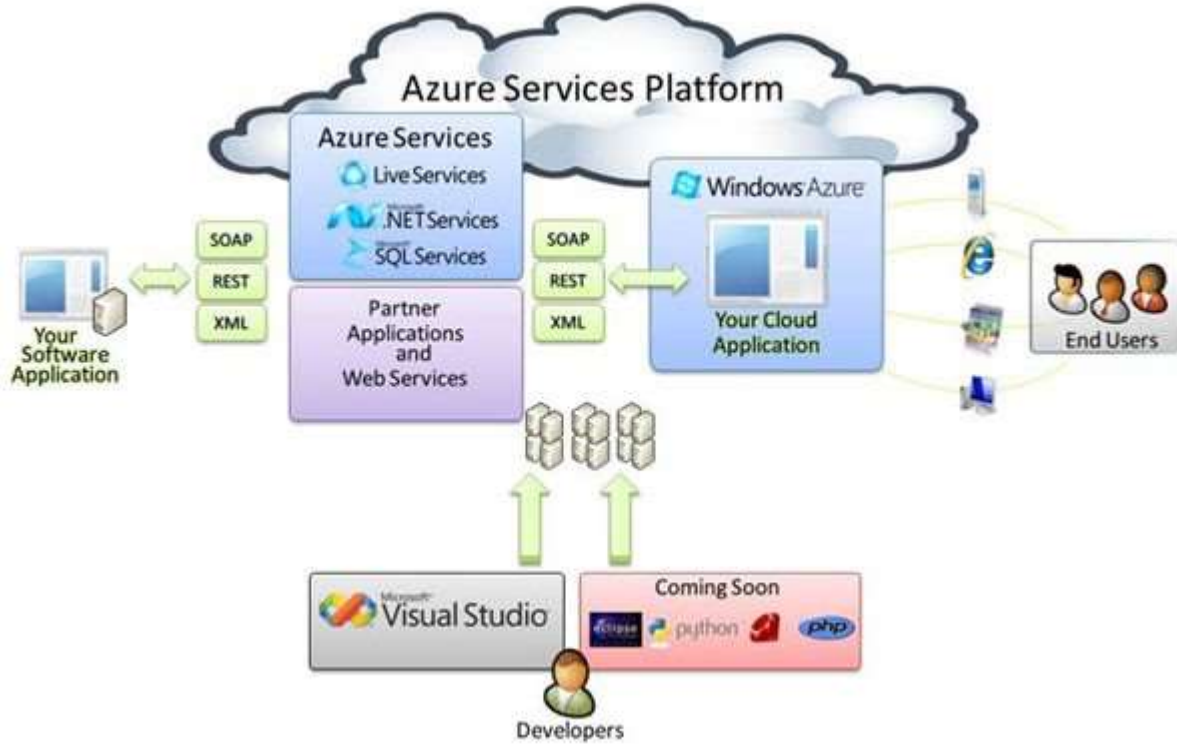
Kıyafet satıř maėaza uygulaması Cloud Computing alt yapsını kullanıldıėı bir mimaride konumlandırıldıėında sunucu satın alma ve kurulma sreci kredi kartınızın bilgilerinizi girmeniz kadar hızlıdır.

İnternet maėazası ierisinde yapılan kampanya srecinde, internet maėazasının sahip olduėu sunucu, bellek, enerji ve gvenlik gereksinimleri bulunmaktadır. Cloud Computing mimarisi ile internet maėazasının gereksnim duyguėu gereksinimlere baėlı olarak otomatik olarak deėiřtirilmektedir. Yapılan deėiřimler sunucu satın alma, kurulum ve diėer iřlemlerin yapılmasını beklemeksizin geniřlemektedir. Sz konusu ihtiyaların saėlanması, kullandıėın kadar deme řeklinde kullanılması, fazla sunucu ve enerji tketilmesini nleyecektir.

Cloud Computing mimari geliřimleri ile ilgili olarak birok zm bulunmaktadır. Sz konusu zmler kendileri ierisinde de destekledikleri platfomlar bulunmaktadır. Ařaėıda bazı Cloud Computing Platfom saėlayacıları bulunmaktadır.

- Microsoft Windows Azure
- Amazon Web Service
- Google App Engine

Cloud Computing Platform sağlayıcıları ile ilgili olarak geçtiğimiz terihlerde San Fransico da düzenlenen Meet Windows Azure konferansı ([MEET Windows Azure konferansı ile ilgili konferans notlarımı bağlantıda bulabilirsiniz.](#)) ile Windows Azure platformunun diğer Cloud Computing platformlarından avantajları ve platform yeniliklerinden bahsedilmiştir.



Geliştirilmiş olan Kıyafet satış mağaza uygulamasının Windows Azure Platform'a taşınması istemiş olduğunuz uygulamanızı geliştirme sürecinde birçok uygulama alt yapısını kullanabilmekteyiz. Aşağıda Windows Azure Platform'un desteklemiş olduğu teknolojiler bulunmaktadır.

- ASP.Net MVC & WebForm
- Ruby
- Python
- NodeJS
- F#
- Java
- Php

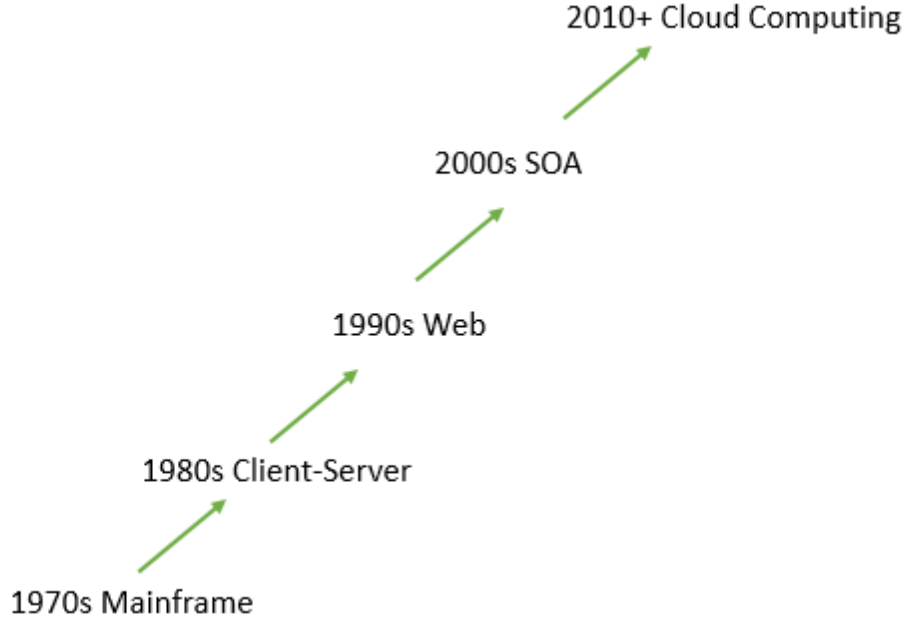
Windows Azure Platform'un uygulama geliştirme işlemleri ile ilgili olarak Eclipse IDE ve Visual Studio IDE araçları ile tam destek sağlamaktadır.

Geliştirmiş olduğumuz uygulama, yukarıda belirtilen uygulama geliştirme teknolojilerini desteklemiyor ise, Virtual Machine desteği sayesinde istemiş olduğunuz uygulama geliştirme mimarisini geliştirerek kullanılmasına olanak sağlamaktadır. Söz konusu özellik sayesinde, Windows Azure Platform'un üzerinde istenilen uygulama kullanıcılara sunulabilmektedir.

Cloud Computing mimarisi, 1970 yılından başlayarak gelişen süreç içerisinde kullanılan alt yapı sistemlerin geliştirilmesi ve eksiklerinin giderilmesi ile ortaya çıkarılmıştır. Bu yazılım ile Cloud Computing mimarisi ve Windows Azure Platform ile ilgili anlatımlar gerçekleştirilmiştir.

EK Not: Bölüm ile ilgili uygulama videosu ve sunumunu incelemek için aşağıdaki bağlantıyı kullanabilirsiniz.

[http://www.ibrahimatay.org/post/2012/12/16/Windows-Azure-ile-Cloud-Computing-Uygulamalari---1-\(-Video-\).aspx](http://www.ibrahimatay.org/post/2012/12/16/Windows-Azure-ile-Cloud-Computing-Uygulamalari---1-(-Video-).aspx)



Windows Azure ile Cloud Computing Uygulamaları – 2

Cloud Computing mimarisi, geliştirici yönünde birçok problemi çözümlenmektedir. Söz konusu çözümler ile birlikte terminolojimize yeni kavramlar ve yaklaşımlar eklenmiş bulunmaktadır. Bu yazımda Cloud Computing sistemleri üzerinde bulunan uygulamaların Scalability süreçlerine değinmiş bulunmaktayım.

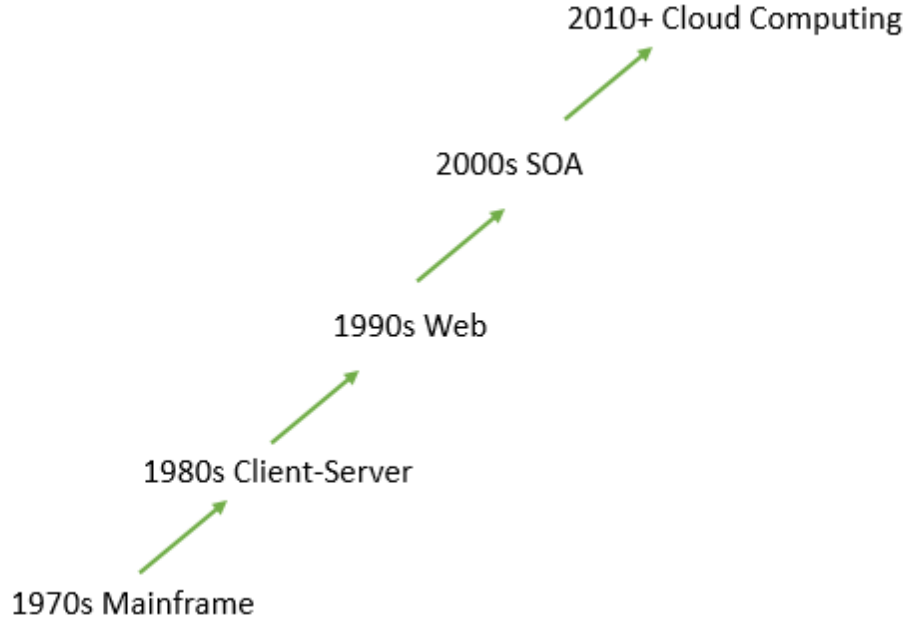


Yazdan kalma bahar günleri yaşamaktayız. Güneş, kimi zaman bulutların arkasına saklanarak, hayatımızda farklılıklar yaratıyor. İşimize geldiğimizde, notlarımızın, görevlerimizin ve çalışmalarımızın yolunda gitmesini umarız. İşlerimizi gerçekleştirirken, bilerek ya da bilmeyerek bulutları kullanırız.

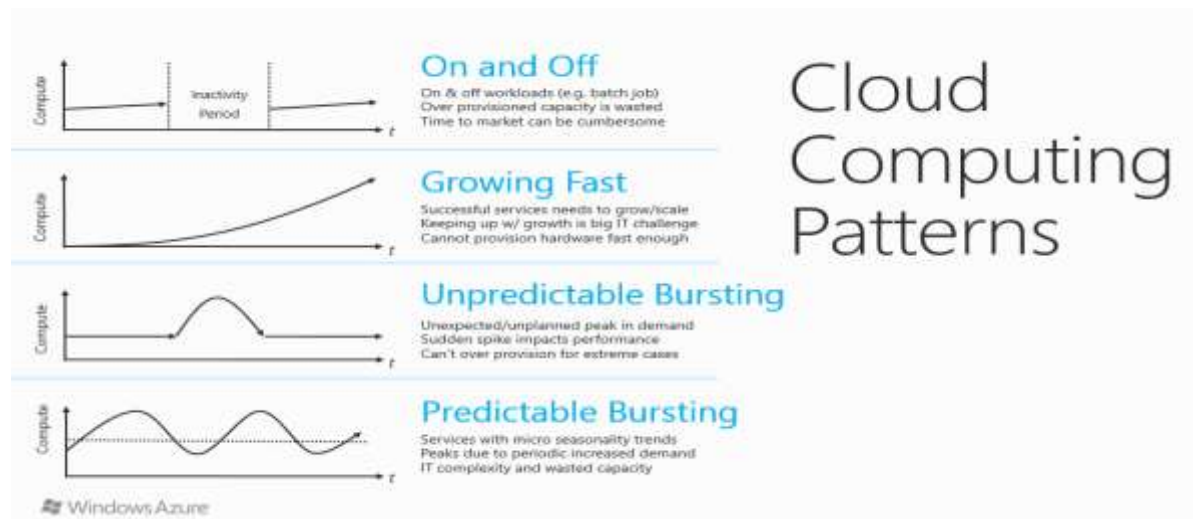


Görevlerimiz yazılım geliştirmek, süreçleri takip etmek ya da takımların organizasyonu sağlamak ise, zaman içerisinde birçok mimari ile iç içe olmak zorundasınızdır. Geçmişte Visual Basic 6 ile geliştirmiş olduğumuz uygulamaları, günümüzde Web uygulamaları olarak görmekteyiz.

Geçtiğimiz tarihlerde yayınlamış olduğum, “Windows Azure ile Cloud Computing Uygulamaları - 1” isimli yayın ile insan hayatı ve bilişim dünyasındaki gelişmelerden bahsedilmişti. Söz konusu süreçler 1970 yılından başlayarak devam etmektedir.



İnsanların ve hizmet vermeyi amaçlayan sistemlerin değişimleri sürekli olarak devam etmektedir. Yaşadığımız dönem içerisinde insanlar istedikleri ortam ya da sistemle Dünya ile bağlantıda kalmak istemektedirler. İsteklere cevap üretilmesi amacı ile yeni uygulama geliştirme mimarileri oluşturulmaktadır. Mimariler, dönemin ihtiyaçlarına uygun olarak çeşitlilik göstermektedir. İçerisinde bulunduğumuz dönem problemlerine bağlı olarak Cloud Computing mimarisi oluşturulmuştur. Cloud Computing mimarisi, uygulama geliştirilmesi ve sunulması ile ilgili olarak birçok konuda kolaylıklar sağlamaktadır.



Cloud Computing uygulamaları için, farklı çalışma tarzları bulunmaktadır. Söz konusu çalışma tarzları, uygulamaların kullanım yoğunluklarına bağlı olarak değişim göstermektedir. Yukarıda belirtilen grafik ile uygulama çalışma tipleri belirtilmiştir. Grafik içerisinde belirtilen uygulama çalışma tarzları ile ilgili açıklama bulunmaktadır.

On and Off

Kurumlar içerisinde kısa dönem de bazı uygulamaların çalıştırılması gerekmektedir. Uygulamalar gerektiğinde çalıştırılıp ve işlem tamamladığında kapatılması ile yaşam süresini tamamlamaktadır. Uygulama sürekli olarak kullanılmadığı için, sistem gereksinimleri kullanmasının sarfiyatıta yol açmaması nedeni ile kapatılmaktadır.

Growing Fast

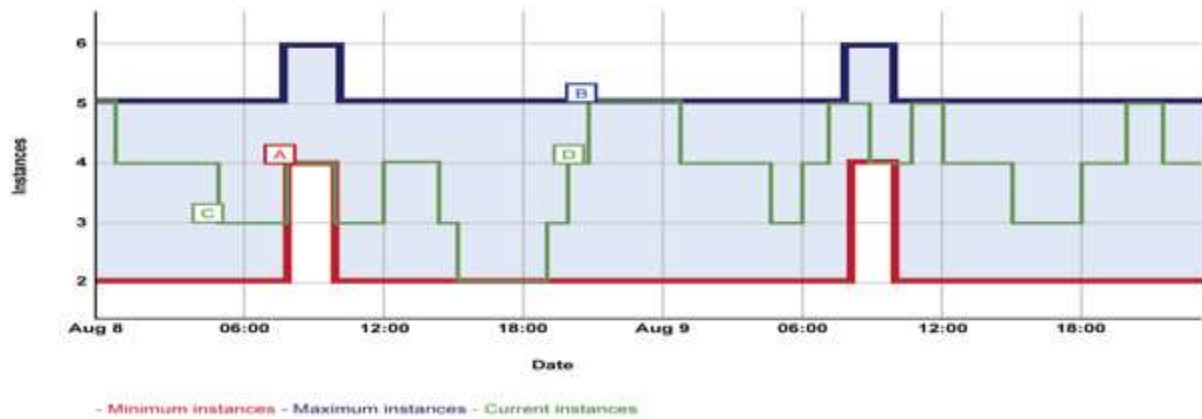
Günümüzde birçok yeni iş fikri oluşturulmaktadır. Genel olarak düşünüldüğünde internet iş fikirleri yoğun olarak artmaktadır. Yeni oluşturulan iş fikri, başladığında bazı nedenlerden dolayı, minimum kullanım düzeyinde çalışmaktadır. İş fikri ile ilgili gerekli çalışmaların yapılması ile kullanımı, artmakta ve artan performans gereksinimleri grafiği oluşmaktadır.

Unpredictable Bursting

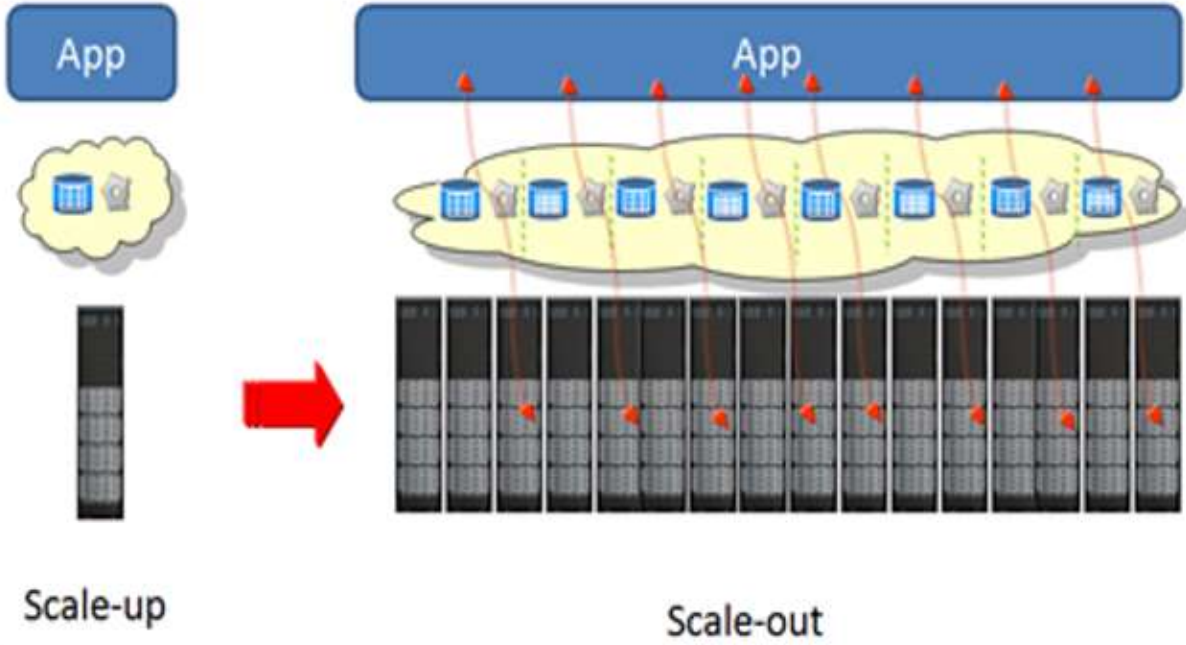
Kurumlar içerisinde mali işlemler, sürekli olarak gerçekleşmektedir. Artan mali bilgilerle, vergi hesaplama dönemlerinde finans uygulamalarının çalışma süreleri artması ve buna bağlı olarak sistem gereksinimleri de artmaktadır. Vergi hesaplama dönemlerinin tamamlanması ile uygulama doğal çalışma sürecine dönerek doğal gereksinimlerine çekilmektedir.

Predictable Bursting

İnternet üzerinde gerçekleşen pazarlama süreçleri, sürekli olarak artmaktadır. İnsanların kişisel zevkleri, istekleri ve yapılan kampanya süreçlerine bağlı olarak, internet alış/veriş sitelerinin de yoğunlukları sürekli olarak değişmektedir. Gerçekleşen süreçte, internet pazarlama uygulamasının farklı sistem gereksinimlerine sahip olması gerekmektedir.



Yaşayan uygulamalar, değişik dönemlerde farklı çalışma yoğunlukları tarzı sergilemektedir. Söz konusu tarzlar uygulamaların yoğunluklarına göre değişim göstermektedir. Uygulamalar üzerinde oluşturulan yoğunluklara bağlı olarak, performans gereksinimleri artmakta ya da azalmaktadır.



Uygulama performansına bağlı olarak, uygulama makine sayılarının artırılması ya da mevcut makinelerin güçlendirilmesi gündeme gelecektir. Söz konusu durumlar uygulama yaşam süresi ile ilgili olarak farklı durumlardır. Söz konusu süreçler yukarıda şekillendirilmiştir.

Cloud Computing mimarisinin kullanmış olduğu, uygulamada birden fazla makine ile işin (Örneğin: Internet portal) gerçekleştirilmesi amaçlanıyor ise, Scale-Out (Makine sayısının artırılması) yapılması gerekmektedir. Sürecin tamamlanması ile Scale-In (Makine sayısının azaltılması) yapılarak uygulamanın doğal sınırlarına ulaştırılabilmektedir.

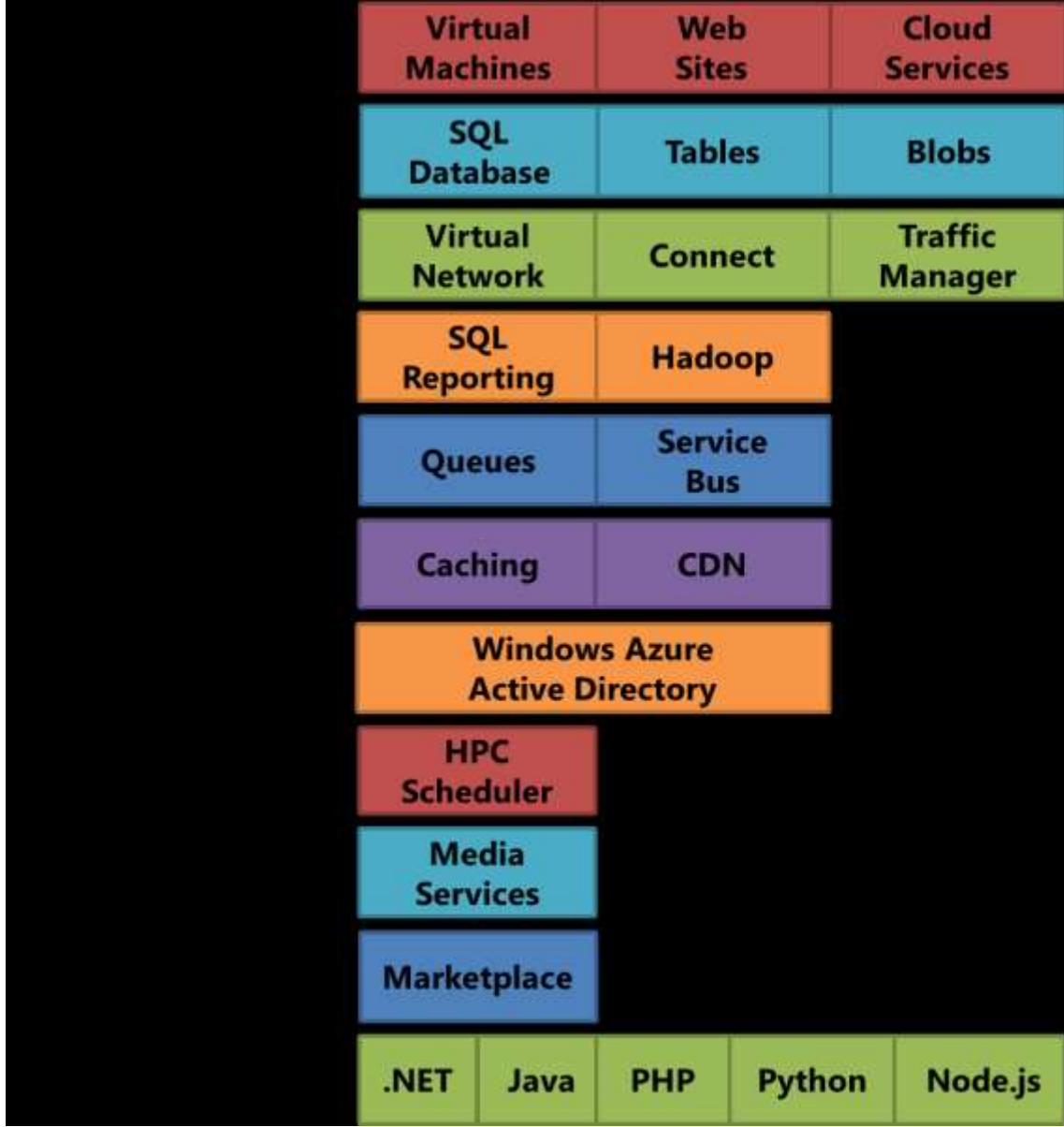
Kullanılan uygulamanın aynı makine üzerinde çalıştırılarak, gerçekleştirilecek olan işin (Örneğin: resim işleme süreçleri) tamamlanması istenildiğinde Scale-Up (Memory ya da CPU artışı) yapılması gerekmektedir. İşlemlerin tamamlanması ile Scale-Down (Memory ya da CPU azaltılması) yapılarak uygulama makinesinin doğal sınırlarına ulaştırılabilmektedir.

Not: Windows Azure mimarisi ile uygulama Scalability'nin otomatik olarak şekillendirilmesi sağlanabilmektedir.

Cloud Computing mimarisi, geliştirici yönünde birçok problemi çözmektedir. Söz konusu çözümler ile birlikte terminolojimize yeni kavramlar ve yaklaşımlar eklenmiş bulunmaktadır. Bu yazımda Cloud Computing sistemleri üzerinde bulunan uygulamaların Scalability süreçlerine değinmiş bulunmaktayım.

EK Not: Bölüm ile ilgili uygulama videosu ve sunumunu incelemek için aşağıdaki bağlantıyı kullanabilirsiniz.

[http://www.ibrahimatay.org/post/2012/12/21/Windows-Azure-ile-Cloud-Computing-Uygulamalari---2-\(-Video-\).aspx](http://www.ibrahimatay.org/post/2012/12/21/Windows-Azure-ile-Cloud-Computing-Uygulamalari---2-(-Video-).aspx)



Windows Azure ile Cloud Computing Uygulamaları – 3

Şirketler iç ve dış operasyonları yönetmek amacı ile çeşitli enstrümanlar kullanmaktadır. Operasyon süreçlerinin değişmesi, uygulama sistemlerinin de değişimlere adapte olabilmesi gerekmektedir. Söz konusu süreç işletmelerin sürekliliğini devam ettirebilmesi için büyük önem taşımaktadır. Windows Azure Platformu, şirketler için birçok kolaylıklar ve minimum maliyet ile şirketlere özgür çalışma imkanı sağlayabilmektedir

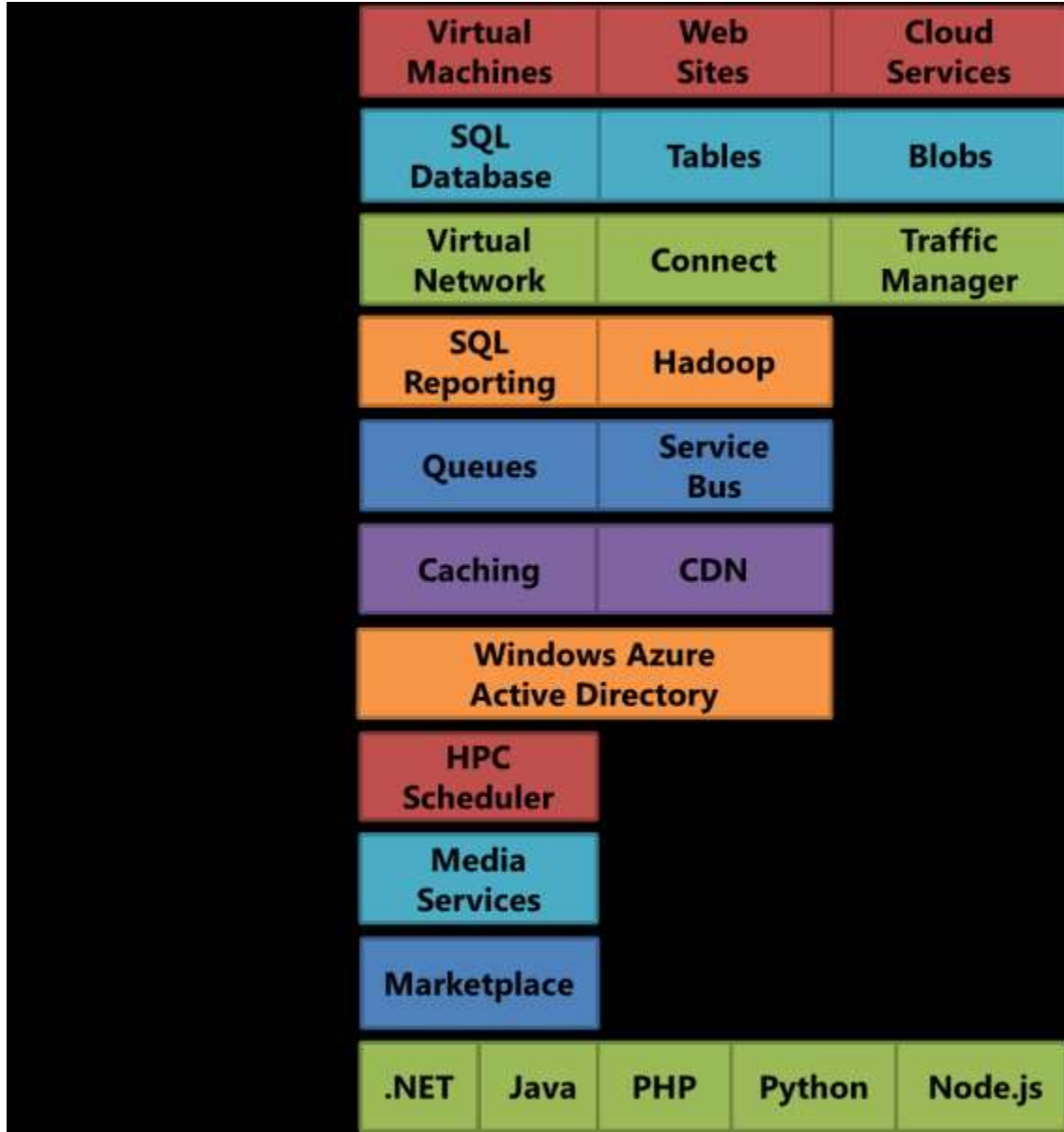


Çalışma yaşamı boyunca farklı ihtiyaçlar ortaya çıkmaktadır. Günümüzde görsel ve işitsel iletişimin ya da eğitim araçlarının artması, hazırlanan çalışmaların sürece uygun olarak tasarlanmasını gerektirdi. Söz konusu süreç; çalışanlar ya da müşteriler ile eğitim videolarının ve dokümanların paylaşılması örnek gösterilebilir. Gerçekleştirilen operasyonlar, şirket içi ya da dışında kullanılan hizmetler ile sağlanmaktadır.

Günümüzde şirketinizin ne kadar büyük olduğu değil, süreçlere ne kadar hızlı adapte olduğunuz ve cevap verdiğiniz önem kazanmaktadır. Büyümeye çalışan şirketler, artan operasyonları ve yoğunluklarına bağlı olarak özgür çalışabilecekleri sistemleri kullanmaktadır.

Şirketler gereksinimlerine bağlı olarak, operasyon süreçlerini kolaylaştırmak amacı ile birçok uygulama kullanmaktadır. Kullanılan uygulamaların kullanımı ve bilginin güvenliğini sağlamak amacı ile çeşitli IT kuralları oluşturmaktadır. Oluşturulan kurallar, şirket stratejilerine bağlı olarak hazırlanmaktadır. Güvenlik kuralları, IT yetkililerinin tecrübe birikimleri ile sınırlı olmaktadır. Kısıtlı IT kurallarına bağlı olarak da kısıtlı kontrol listeleri ve gözden kaçırılan güvenlik zafiyetleri oluşmaktadır. **Cloud Computing** sağlayıcıları ve özellikle **Windows Azure platformu ISO/IEC 27001** standartları çerçevesinde güvenlik süreçleri yönetmektedir.

ISO/IEC 27001 standartları, *International Electrotechnical Commission* ve *International Organization for Standardization*, kurumların risk yönetimi ve risk işleme planlarını, görev ve sorumluluklarını, iş devamlılığı planlarını, acil durum olay yönetimi prosedürleri hazırlamasını ve uygulamada bunların kayıtlarının tutmasını süreçlerini incelemektedir.



Şirketler operasyon süreçlerini yönetmek ve izlemek amacı ile birçok çözümler kullanmaktadır. Kullanılan çözümler operasyon süreçlerinin büyümesi ile yetersiz kalmasına neden olmaktadır. Artan uygulama gereksinimleri yeni sunucu, uygulama lisansları ve yöneticilerin eğitilmesi ya da işe alınması gereksinimlerini doğurmaktadır. Gelişen şartlar şirketlerin, gereksinimlerine bağlı olarak gereksinim çözümleri oluşturmasına neden olmaktadır. Yapılan planlamalar ile ilgili olarak birçok senaryo oluşturulabilmektedir. Aşağıda örnekte bazı gereksinim senaryoları bulunmaktadır.

Senaryo - I

Şirket içerisinde çok miktarda video içerik bulunmaktadır. İçerikleri müşterilere ya da çalışanlara nasıl ulaştırabiliriz?

Uygulanan klasik Çözüm - I

Şirketler oluşan gereksinime göre satın almalar yapmaktadır. Şirketin kendisi içerisinde ya da dışarıya açık video yayını yapabilmek amacı ile donanım, işlemlerin yerine getirme amacı ile yazılımların satın alınması gerekmektedir. Yapılan satın alma işlemlerine bağlı olarak, sistemlerin yönetimini gerçekleştirecek takım elemanlarının eğitimi ya da işe alınması gerekmektedir. Gerçekleşen sürecin, zaman içerisinde artan verilerin depolanması, güvenliği gibi nedenlerden dolayı yeniden şekillendirilmesi gerekmektedir.

Windows Azure Platform'u ile Uygulanabilecek Çözüm

Windows Azure Platform kullanılması ile şirket gereksinim ve stratejilerine bağlı olarak, video içeriklerin şekillendirilmesi ve kullanıcılar için yayınlanabilmesi mümkündür. Süreci gerçekleştirmek amacı ile **Windows Azure Media Service** ile ilgili birkaç bilgi öğrenilmesi yeterli olmaktadır. Kullanılan **Windows Azure platform** hizmetleri gereksinimlere bağlı olarak istenildiği gibi şekillendirilebilmektedir.

Senaryo - II

Şirketler pazarlama süreçlerini başarıya ulaştırmak amacı ile müşterilerini tanıyarak, **CRM** süreçlerini gerçekleştirmektedir. Şirketlerin **CRM** süreçlerini gerçekleştirmek için, müşteri ile ilgili verileri depolamak ve gerektiğinde analiz yapması gerekmektedir.

Uygulanan klasik Çözüm - II

Şirketlerin hedef kitlelerine ulaşmak amacı ile müşterilerinin gerçekleştirmiş olduğu satın alma ya da zevkler ve ilgilendikleri konular ile ilgili bilgiler toplaması gerekmektedir. Büyümekte olan şirketlerin, artan müşteri portföyüne bağlı olarak toplaması gereken bilgi de artmaktadır. Artan bilgilere bağlı olarak bilgilerin depolanması, analiz edilmesi ve pazarlama stratejisi oluşturma süreçleri, alt yapıların yetersiz kalması nedeni ile yavaşlayacak ve istenilen pazarlama manevralarının yapılmasını engelleyecektir. Operasyon süreçlerinin devamlılığını sağlanabilmesi için yeni alt yapı sistemlerinin satın alınması gerekmektedir.

Windows Azure Platform'un ile Uygulanabilecek Çözüm

Şirketlerin hedef kitlelerine, ulaşılabilmesi amacı ile kitlenin tanınması ve kitlelere bağlı olarak stratejilerin hızlı şekilde oluşturulması gerekmektedir. Şirket, artan veri havuzunu **Windows Azure**'a taşıması ile **Big Data (Hadoop, MongoDB)** çözümleri ya da **SQL Database** seçenekleri kullanarak depolayabilmektedir. Depolanan veriler, **SQL Reporting** hizmeti ile hızlı şekilde raporlanabilmektedir. Söz konusu işlemler ve gereksinimler birkaç küçük konfigürasyon ile şekillendirilebilmekte ve süreç devam etmektedir.

Senaryo - III

Şirket içerisinde gerekli olan operasyonların gerçekleştirilebilmesi amacı ile çeşitli gereksinimlere sahip uygulama makineleri gerekmektedir. Söz konusu makinelere farklı lokasyonlar üzerinden erişebilmesi gerekmektedir.

Uygulanan klasik Çözüm - III

Operasyonların devamını sağlamak amacı ile farklı seçeneklerin sağlanabildiği donanım gereksinimlerinin satın alımlarının yapılması gerekmektedir. Satın alınan ekipmanların yönetilebilmesi amacı ile gerekli yazılım alınması ve yönetim süreçleri gerçekleştirecek yöneticilerin eğitilmesi ya da işe alınması gerekmektedir.

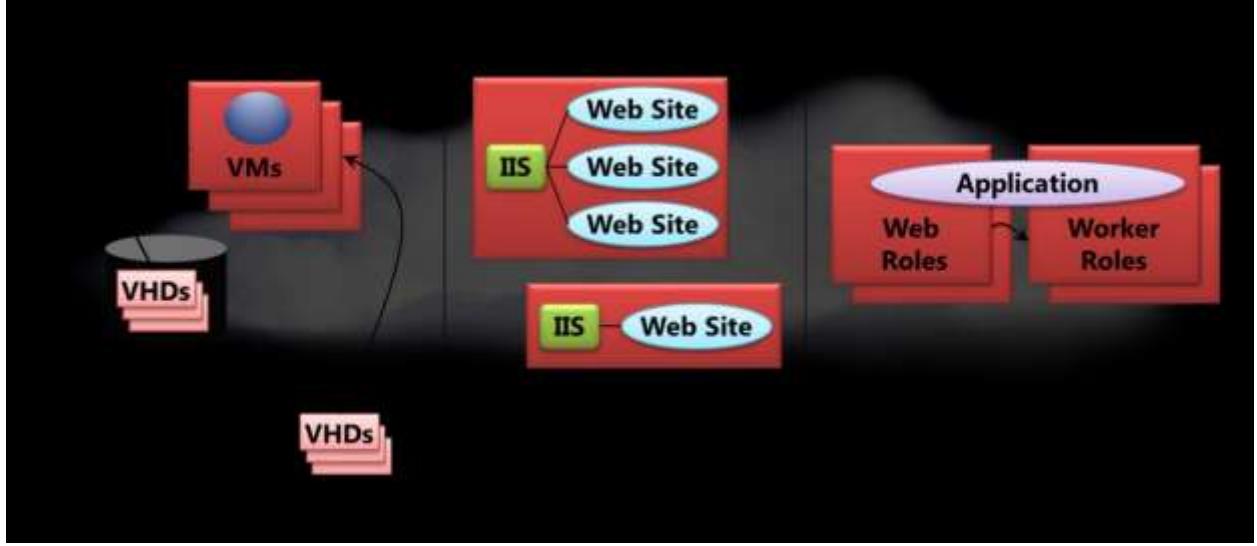
Windows Azure Platform'un ile Uygulanabilecek Çözüm

Şirketler kullandıkları teknolojik enstrümanlara göre farklı konfigürasyona ve işletim sistemine sahip makineler kullanmaktadır. Özellikle kampanya dönemine girecek olan şirketler için operasyonlarını yönetebilmesi için gerekli alt yapıların sağlanması büyük önem taşımaktadır. Şirketler sahip olduğu alt yapı istemleri **Windows Azure Platformu**'na taşınması ile istenilen konfigürasyon ve sayıda makinelere sahip olabilmektedir. Özellikle şirket isteklerine bağlı olarak Windows, Linux ya da farklı yazılım sistemlerini barındırabilmektedir.

Şirketler iç ve dış operasyonları yönetmek amacı ile çeşitli enstrümanlar kullanmaktadır. Operasyon süreçlerinin değişmesi, uygulama sistemlerinin de değişimlere adapte olabilmesi gerekmektedir. Söz konusu süreç işletmelerin sürekliliğini devam ettirebilmesi için büyük önem taşımaktadır. **Windows Azure Platformu**, şirketler için birçok kolaylıklar ve minimum maliyet ile şirketlere özgür çalışma imkanı sağlayabilmektedir.

EK Not: Bölüm ile ilgili uygulama videosu ve sunumunu incelemek için aşağıdaki bağlantıyı kullanabilirsiniz.

[http://www.ibrahimatay.org/post/2013/1/15/Windows-Azure-ile-Cloud-Computing-Uygulamalari---3-\(-Video-\).aspx](http://www.ibrahimatay.org/post/2013/1/15/Windows-Azure-ile-Cloud-Computing-Uygulamalari---3-(-Video-).aspx)



Windows Azure ile Cloud Computing Uygulamaları – 4

Uygulama geliştirilmesi ya da mevcut uygulamaların Windows Azure Platform 'a taşınması ile ilgili birçok senaryo üretilebilir. Geliştirilen senaryolar; zaman, maliyet ya da süreçlerin basit olarak gerçekleştirilebilmesi mümkün olmaktadır. Örnek verilen senaryolar, istek ve mevcut koşullara bağlı olarak şekillendirebilecektir. Windows Azure Platform 'u ile şirketler minimum maliyet ile özgür çalışmalarına olanak sağlamaktadır.



Zamanın geçtiğini, yaşam içerisinde meydana gelen değişimler ile fark etmekteyiz. Gün güneş ışığı ile yeni başlangıçlar hazırlanmakta, yeni başlangıçlar sahip olduğumuz alışkanlıkların değişmesine neden olmaktadır. Zaman insanlara adapte olmayı ve ilerlemeyi öğretmektedir.

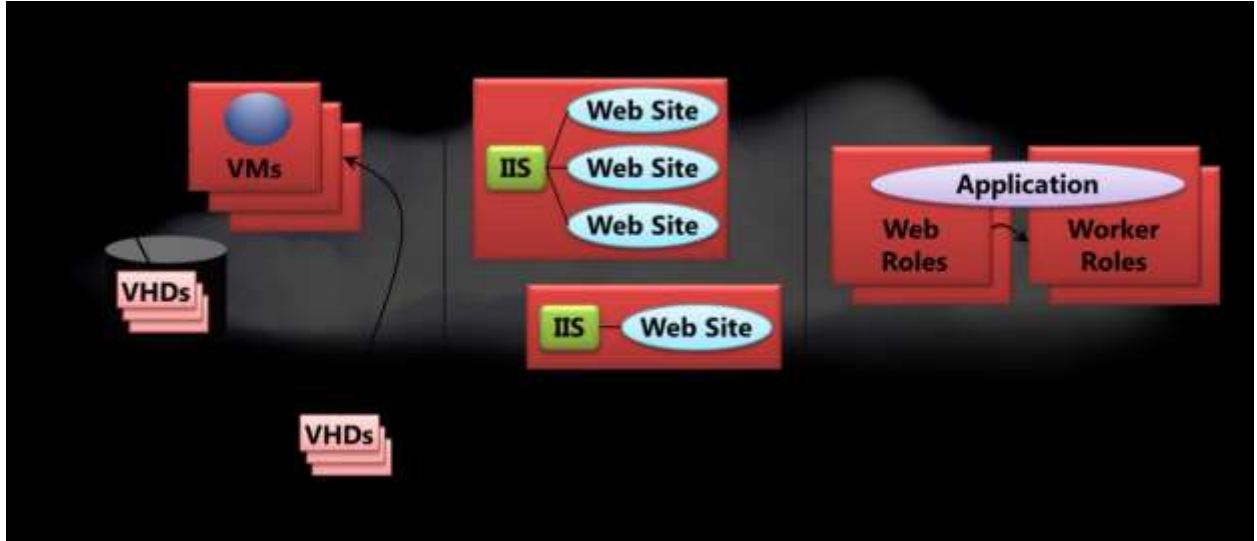
İnsanlar değişmekte ve değişimlerine en büyük katkıyı, geliştirilen uygulamalar ve kullanım süreleri etki etmektedir. Geçmişte konuşmak için kilometrelerce uzaklıkları aşmak için uğraşırken, artık bilgisayarın diğer yanında görebiliyoruz. Günlük hayatta kullandığımız uygulamalar hayatlarımızı değiştirdi. Meydana gelen değişim uygulama geliştiricilerinin uygulama geliştirme süreçlerini etkiledi ve süreçlere hızlı adapte olmasına neden oldu.

Uygulama geliştirme süreçleri, geliştirilecek uygulama türünü, geliştirme aşamasında kullanılacak olan Framework ve takım kültürüne göre farklılıklar göstermektedir. Günümüzde Web uygulamalarının geliştirilmesi ve kullanıcı yoğunluklarının artması ile uygulamaların Windows Azure ve benzeri hizmet sağlayıcı sistemlere taşınmasına yönelik çözümler oluşturulmaya başladı.

Günümüzde Cloud Computing sağlayıcılara yönelik geliştirilen uygulamalar, sağlayıcı şirketin yaşam süresine ilişkili olarak yaşamını devam ettirmektedir. Cloud Computing sağlayıcılar ve özellikle Windows Azure Platform, sağlamış olduğu Cloud Computing hizmetlerini European Commission ve benzeri kurumların standart çalışmalarına bağlı olarak şekillendirmektedir.

Cloud Computing üzerinde barındırılacak olan uygulamaların, standartlarına bağlı olarak geliştirilmesi Cloud Computing sağlayıcı şirketlerin de bağımsız olarak, uygulamaları istediği sağlayıcı şirkette konumlandırmasına olanak sağlanmaktadır.

bölümü ile şirketler için **Cloud Computing** altyapısının önemi ve **Windows Azure Platform** ile kazanacakları bazı olanaklar hakkında bilgi verilmiştir.



Uygulama geliştirme süreçlerinde, geliştirilmesi istenen uygulamaya bağlı olarak, farklı uygulama gereksinimleri ortaya çıkabilmektedir. Gereksinimler uygulama geliştirme

aşamasında başlayarak, uygulama yaşam döngüsünün tamamlanması sürecin de devam etmektedir. Geliştirme süresini tamamlanmış olan uygulamalar, yaşamları boyunca gereksinimlere bağlı olarak farklı süreçlere dahil edilebilmektedir. Söz konusu süreçler uygulamanın Cloud Computing sağlayıcısına taşınması ya da Cloud Computing odaklı olarak geliştirilmesi şeklinde senaryolaştırılabilmektedir. Aşağıda uygulamaların Cloud Computing sağlayıcılarından olan Windows Azure Platform 'una örnek taşıma senaryoları belirtilmiştir.

Senaryo – I

Geliştirme süreci tamamlamış ve çalışmakta olan uygulamalarımız bulunmakta. Uygulamaları Windows Azure Platform 'una taşımak istiyoruz. Fakat uygulamalar ile ilgili yeni geliştirme yamamıza zamanımız bulunmamaktadır. Uygulamalarımızı Windows Azure Platform 'una nasıl taşıyabiliriz?

Çözüm Önerisi – I

Taşınması istenen uygulamaların, Windows Azure Platform 'un sağladığı kolaylıklarında biri olan sanal makine üzerine konumlandırarak yaşam sürecine devam ettirilebilir.

Senaryo – II

Çalışmakta olan uygulamalarımız bulunmakta. Uygulamalarımız Linux üzerinde çalışmakta ve Windows Azure Platform 'un kullanmak istiyoruz. Fakat uygulamalarımızın kullanmış olduğu veritabanı ve uygulama mimarisini (programlama dili ya da 3rdParty yazılımlar) Windows Azure Platform desteklememekte. Uygulamalarımızı Windows Azure Platform 'una taşımak için ne yapmamız gerekmektedir?

Çözüm Önerisi – II

Windows Azure Virtual Machine özelliği ile uygulamalar için istenen yaşam ortamı oluşturulabilir. Windows Azure Virtual Machine desteği ile Linux (CentOS,Ubuntu ve Suse), Windows Server(2008 ve 2012) işletim sistemleri kullanılabilir. Taşınması istenen uygulamaların gereksinimi olan 3rdParty yazılımlar ise, Windows Azure Virtual Machine bağlanılarak kurulumları yapılabilmektedir. Çalışmaların gerçekleştirilmesi ile istenilen uygulamalar Windows Azure Platform 'una kolaylık ile taşınabilmektedir.

Senaryo – III

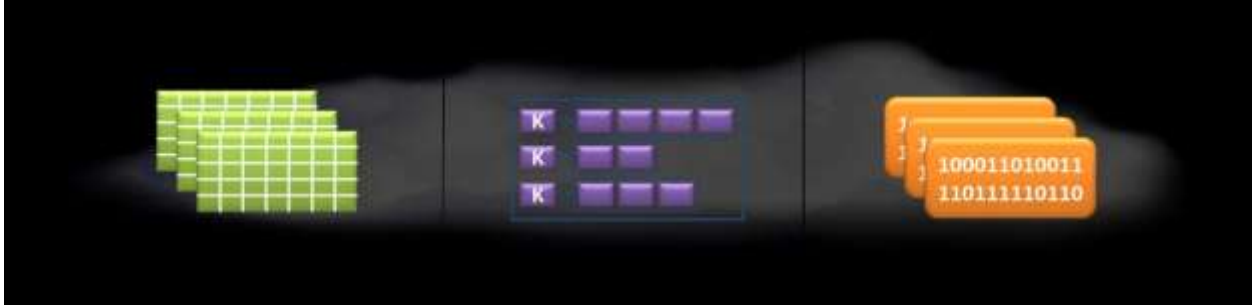
Uygulamalarınızı çalışmakta olduğu mevcut sanal makineler (Hyper-V ya da VMware üzerinde) bulunmakta. Sanal makinelerin bulunduğu donanımlar yetersiz ve zaman ile gereksinimler artmakta. Artan gereksinimleri hızlı ve düşük maliyet ile çözmek istiyoruz. Süreci nasıl çözebiliriz?

Çözüm Önerisi – III

Kullanımda bulunan sanal makineler Windows Azure Platform'a taşınması ile düşük maliyetli ve artan ihtiyaçlara cevap üretebilen sistemler oluşturulabilmektedir.

Kullanımda bulunan sanal makineler Windows Azure Virtual Machine üzerinde taşınarak, basit ve hızlı şekilde işlemler gerçekleştirilebilmektedir.

Belirtilen taşıma senaryoları, uygulama süreçlerinde kullanılması muhtemel önerilerdir. Bu öneriler taşınması istenen uygulama ve izlenmesi gereken stratejilere bağlı olarak değişiklikler ve farklı uygulanmalar olabilir.



Uygulama geliştirme süreçleri, teslim süreleri ya da uygulama geliştirecek olan takıma göre farklılıklar göstermektedir. Özellikle Cloud Computing odaklı uygulama geliştirme süreçlerinde farklı senaryolar üretilebilmektedir. Aşağıda bazı Windows Azure Platform yönelik uygulama geliştirme senaryo örnekleri belirtilmiştir.

Senaryo – I

Geliştirmeye başlayacağımız olan uygulamanın teslim tarihi çok yakın ve eğitim alma zamanımız bulunmamakta. Uygulamamızı Windows Azure Platform odaklı olarak geliştirmek istiyoruz. Uygulamamızı nasıl geliştirebiliriz?

Çözüm Önerisi – I

Klasik uygulama geliştirme süreçlerinde SQL Server veritabanı, verilere performanslı erişebilmek için .Net Cache ya da 3rdParty olarak MemCached kullanılmaktadır. Windows Azure Platform odaklı uygulama geliştirme sürecinde SQL Database ve geliştirilen uygulamaya bağlı olarak Hadoop ya da MongoDB gibi NoSQL veritabanları kullanılabilir. Uygulama Caching süreçleri ile ilgili olarak ise, MemCached kullanabilmemize olan sağlamaktadır. Sağlanan kolaylıklar ile mevcut uygulama geliştirme bilgilerini kullanarak Windows Azure Platform 'un da uygulamalarımızı konumlandırabilmekteyiz.

Senaryo – II

Geliştirmek istemiş olduğum uygulamayı Windows Azure Platform da en iyi performans ve üst düzey güvenlik ile çalışmasını istiyorum. Uygulamamı nasıl geliştirmeliyim?

Çözüm Önerisi – II

Geliştirilmesi istenen uygulamayı, üst düzey güvenlik seviyesinde çalıştırılması amaçlandığı senaryolarda Windows Azure Cloud Services ve Windows Azure 'un veri yığın nesneleri olan Table ve Blob nesneleri kullanılması önerilmektedir. Gerçekleştirilen geliştirme Windows Azure Platform enstrümanlarını kullanarak yapılmış olacaktır.

Uygulama geliştirilmesi ya da mevcut uygulamaların Windows Azure Platform 'a taşınması ile ilgili birçok senaryo üretilebilir. Geliştirilen senaryolar; zaman, maliyet ya da süreçlerin basit olarak gerçekleştirilebilmesi mümkün olmaktadır. Örnek verilen senaryolar, istek ve mevcut koşullara bağlı olarak şekillendirilebilmektedir. Windows Azure Platform 'u ile şirketler minimum maliyet ile özgür çalışmalarına olanak sağlamaktadır.

EK Not: Bölüm ile ilgili uygulama videosu ve sunumunu incelemek için aşağıdaki bağlantıyı kullanabilirsiniz.

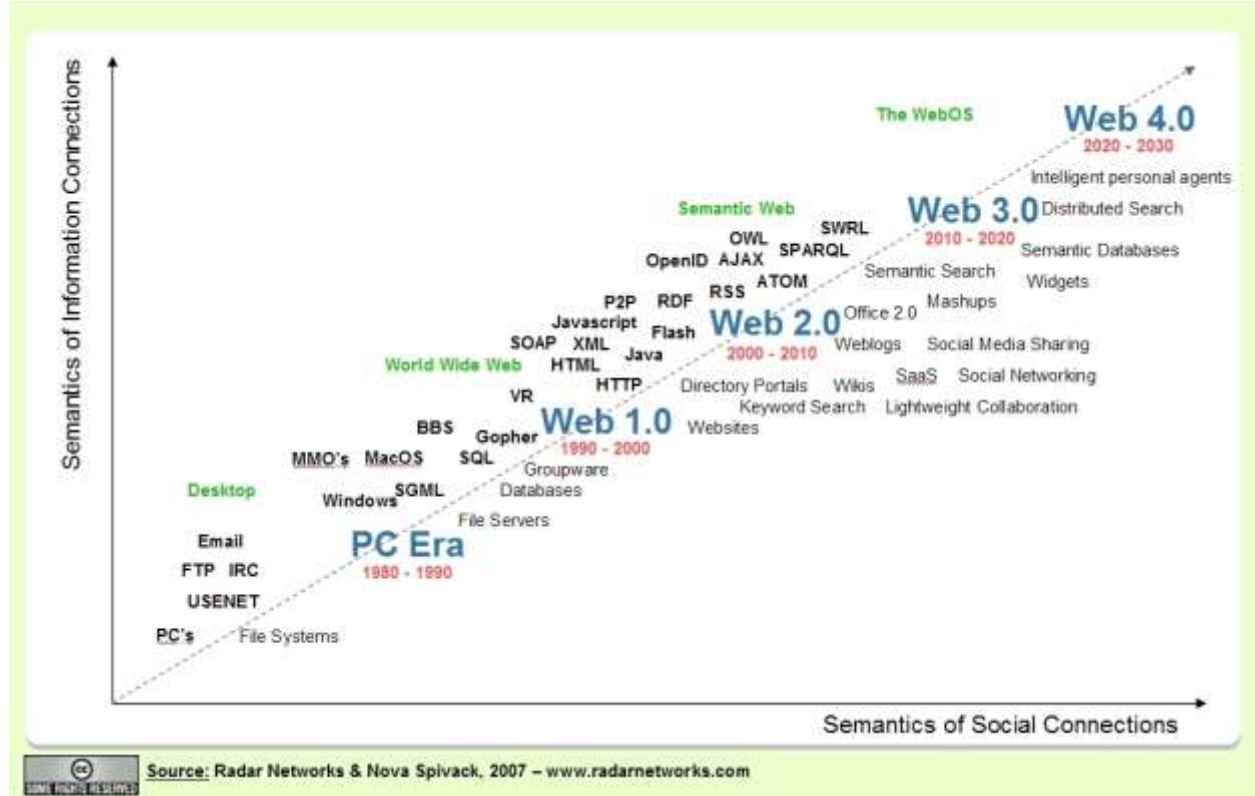
[http://www.ibrahimatay.org/post/2013/1/16/Windows-Azure-ile-Cloud-Computing-Uygulamalari---4-\(-Video-\).aspx.aspx](http://www.ibrahimatay.org/post/2013/1/16/Windows-Azure-ile-Cloud-Computing-Uygulamalari---4-(-Video-).aspx.aspx)

Gün içerisinde birçok yeni konu okuyoruz. İsimize yarayan konuları, kendimize katmaya çalışıyoruz. Her yeni konu, hayatımıza yeni alışkanlıklar ve bakış açıları kazandırmaktadır. Kazanılan yeni alışkanlıklar ve bakış açıları yeni değişimler oluşturmaktadır.

Yeni konuları öğrenmek, eğlenceli olsa da uygulama sürecinde zorlu, ağırdalı ve sancılı olmaktadır. Her yeni konu başlanacak yeni bir nokta olarak düşünülse de mevcut olan bilgilerin üzerinde çıkılan kat olarak görülmesi gerekmektedir. Her yeni oluşturulan kat, zirveye giden yeni bir merdiven olarak görmeliyiz.

Değişim süreçleri, değişime karşı dirençleri oluşturur. Meydana gelen direnç ise, adaptasyon sürecinde yeni adımların atılmasına neden olmaktadır.

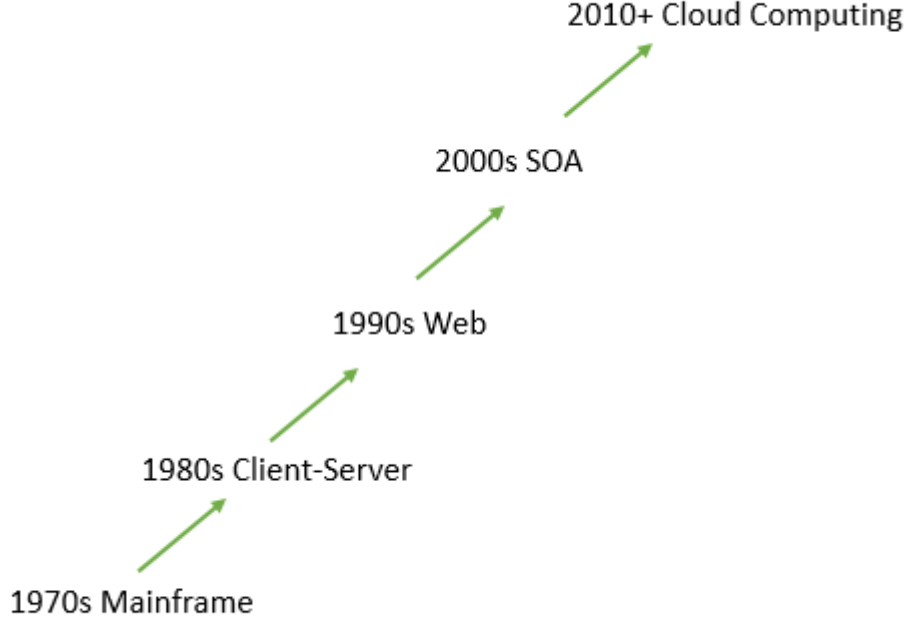
Günümüzde teknolojik yenilikler birkaç gün, hafta ya da aylık değişimler ile gerçekleşmektedir. Sürece geliştirici gözünde bakıldığında ise, son kullanıcıdan daha hızlı adapte olmamız ve fark yaratmamız gerekmektedir.



Geçmişte **Visual Basic** ve **Classic ASP** ile geliştirdiğimiz uygulamaların yerini, **ASP.Net MVC** ya da **SharePoint** 'in kullanılarak geliştirilen projeler aldı. Zaman hızla ilerliyor. Geçen zaman içerisinde değişimleri fark ederek, yeni bilgilere sahip olarak yürünmek gerekmektedir. Süreç, geleceğe daha sağlam adımlar ile yürümeyi sağlayacaktır.

Her değişim süreci, yeni adaptasyon süreçleri oluşturmaktadır. Geliştirici olarak, kullanılan **Framework** yapılarının kısa dönemler ile yeni sürümlerinin yayınlanması, teknik süreçlere hızlı adapte olmamıza neden olmaktadır. Teknik adaptasyon süreçlerinin hızlı, basit ve anlaşılır olabilmesi amacı ile uygulama ortamları hazırlamaktayız. Hazırlanan

uygulama ortamları kimi zaman geçmişte gerçekleştirilen projelerden ya da yeni oluşturulan uygulama senaryoları ile şekillenmektedir.



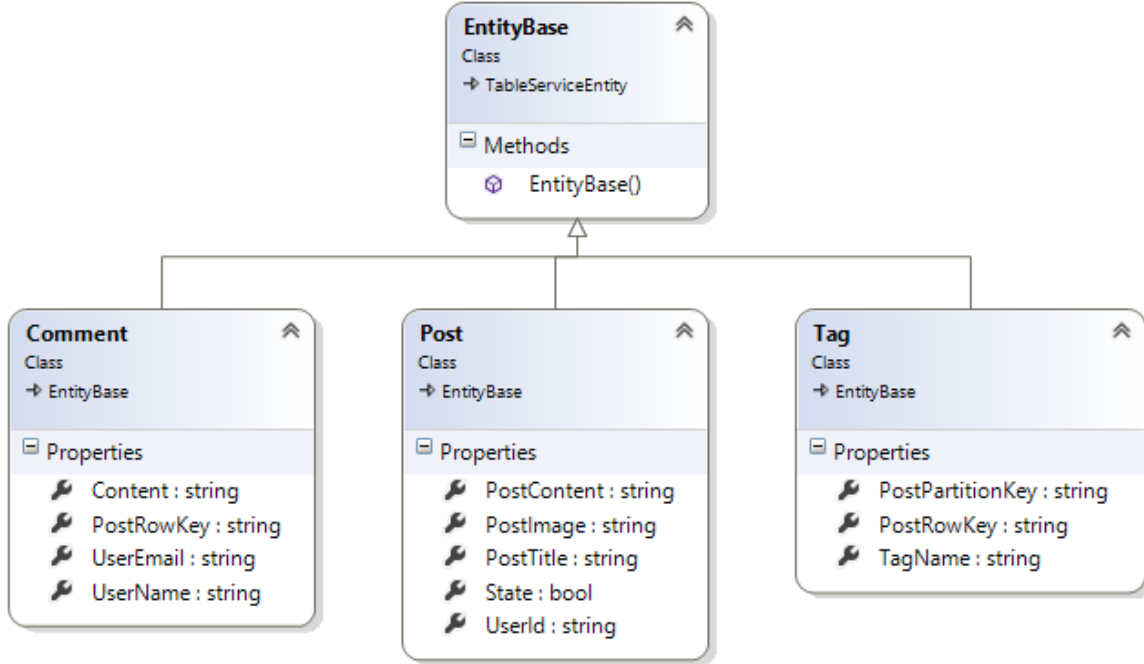
Teknik adaptasyon süreçlerinde hızlı, anlaşılır ve eğlenceli olması amacı ile uygulama projeler geliştirmekteyiz. **Cloud Computing** mimarisi ve özellikle **Windows Azure Platform** anlam amacı ile uygulama projesi geliştiriyor olacağız.

Windows Azure Platform ‘u anlama ve ona eğlenceli şekilde adapte olabilmemiz amacı ile “WindowsAzure.FunnyApp” isimli uygulama projesini geliştiriyor olacağız.

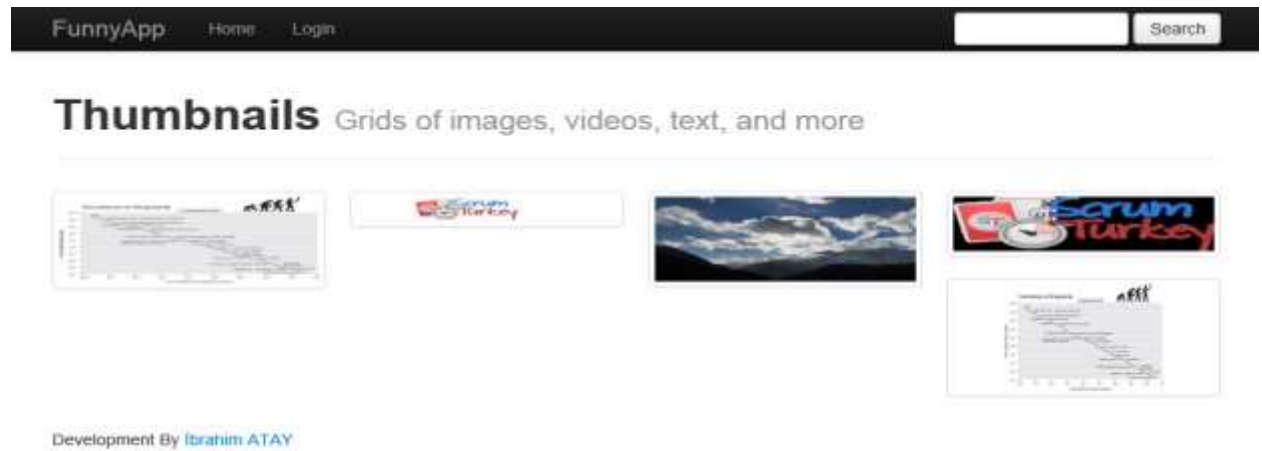
Geliştirilmesi amaçlanan uygulama projesi, günlük hayatımızın parçası haline gelen “Instagram” projesine benzemektedir. Gerçekleştirilen proje ile kullanıcıların diledikleri fotoğrafları uygulama vasıtası ile paylaşması amaçlanmaktadır. Proje içerisinde bazı senaryolar aşağıda belirtilmiştir.

- Kullanıcı hesabının işlemleri (kullanıcı adı, parola, eposta)
- Resim yükleme alanının oluşturulması (istenen resmin yüklenmesi, açıklama, ilgili etiketler)
- Yüklenen resimlerin thumbnail boyutlarında şekillendirilmesi (**Windows Azure Worker Role**)
- Yüklenen resimlerin görüntülenmesi (thumbnail boyutunda resimlerin listelenmesi)
- Resminde detay gösterilmesi (açıklama, etiketler ve yüklene boyutlarda resim)
- Yorum giriş alanının oluşturulması (kullanıcı adı, eposta ve yorum)
- Yorumların listelenmesi (Kullanıcı adı ve yorum gösterilmesi)

Belirtilen uygulama senaryolarına bağlı, veri yapılarının oluşturulması gerekmektedir. Aşağıda uygulama senaryolarının gereksinimlerinin sağlanması amacı ile oluşturulan sınıfın yapısı bulunmaktadır.



Geliştirilmesi amaçlanan uygulama projesi içerisinde kullanıcıların işlemleri gerçekleştirmesi amacı ile **Windows Azure Web Role** (Web uygulaması), resim işleme ve diğer yoğun zaman gerektiren işlemlerin gerçekleştirilmesi amacı ile **Windows Azure Worker Role** (**Windows Service** nesnelerini benzer çalışma prensipleri olan **Windows Azure Role** yapısıdır.) kullanılacaktır. Uygulama veri işlemleri ve katmanlar arasında haberleşme işlemlerini, **Windows Azure Platform** nesneleri olan **Blob**, **Table** ve **Queue** nesnelerini kullanacaktır.



Uygulama projesi **Github** ve **Codeplex** üzerinde bulunan repository üzerinde güncel kodları paylaşılmaktadır. Uygulama projesi ile ilgili olarak yaşanan problemleri çözümlenmesi amacı ile github ya da Codeplex üzerinde bulunan Wiki kullanmanızı önermekteyim. Uygulama repository adresleri aşağıda belirtilmiştir.

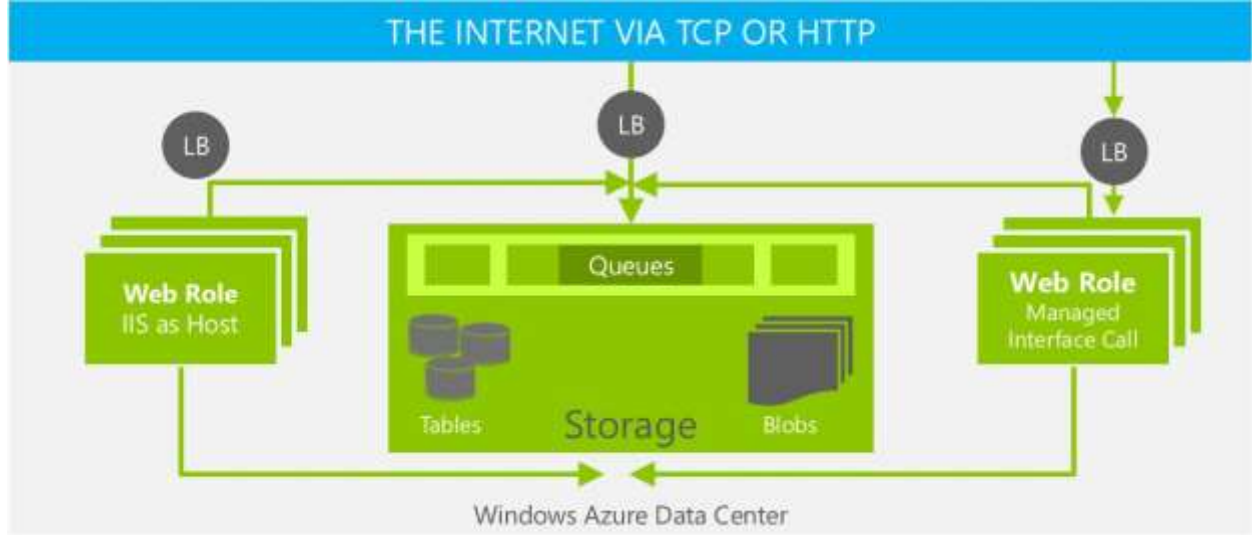
Codeplex / <https://windowsazurefunnyap.codeplex.com/>

Github / <https://github.com/ibrahimatay/WindowsAzure.FunnyApp>

Mesleki gerekliliklere bağlı olarak şekillenen hayatlarda, adaptasyon öğrenilmesi zor olmasına karşın, çözümleri de yanında getirmektedir. Yeni bir adaptasyon sürecine girmektedir. Ama geçmişte sahip olduğumuz bilgilerin üzerine yeni kat oluşturmaktayız. "WindowsAzure.FunnyApp" projesi ile **Windows Azure Platform** ve **Cloud Computing** mimarisini anlayarak. Eğlenceli olarak sürece adapte olmayı amaçlamaktayız.

EK Not: Bölüm ile ilgili uygulama videosu ve sunumunu incelemek için aşağıdaki bağlantıyı kullanabilirsiniz.

[http://www.ibrahimatay.org/post/2013/1/18/Windows-Azure-ile-Cloud-Computing-Uygulamalari---5-\(-video-\).aspx](http://www.ibrahimatay.org/post/2013/1/18/Windows-Azure-ile-Cloud-Computing-Uygulamalari---5-(-video-).aspx)



Windows Azure ile Cloud Computing Uygulamaları – 6

İş uygulamaların başarılı ve performanslı çalışması, uygulama yazılım mimarisi ve konumlandırıldığı platform ile ilişkili olarak gerçekleşmektedir. Windows Azure Cloud Service uygulama geliştirme modeli ile geliştirilen uygulamalar, iş sürecine bağlı olarak parçalarına ayrılarak, beklenen başarı ve performans elde edilebilmektedir.



Şirketler, yaşamlarını sürdürebilmek amacı ile çeşitli iş süreçleri gerçekleştirmektedir. İş süreçleri, sürekli ve başarılı sonuçlar üretilmesi, iş kural ve standartları ile sağlanmaktadır. Oluşturulan iş kuralları, kalitenin korunmasını amaçlamaktadır.

Günümüzde müşteri memnuniyetinin, devamlı ve istenildiği gibi sağlanması, iş uygulamaları ile gerçekleştirilmektedir. İş standart ve kurallarının farklılık göstermesi, kullanıcıları özelleştirilmiş uygulamalara yöneltmektedir.



Müşteri portföylerine, uygun üretim ve hizmetlerin sunulması, iş süreçlerinin sürekli ve başarılı olması ile sağlanmaktadır. İstenilen sonuçlar, süreçleri gerçekleştiren ve izleyen uygulamaların sağlıklı çalışması ile sağlanmaktadır.

İş uygulamaları kullanımı, gerçekleştireceği iş sürecinin yoğunluğuna bağlı olarak değişmektedir. Örneğin; yılbaşı dönemini de içerisine alan yılın son üç ayı, e-ticaret ve şirketlerin muhasebe bölümleri, hesapları kapatabilmek ve ürün satmak amacı ile yoğun olarak çalışmaktadır. Gerçekleştirilen işlemler, yılın diğer zamanlarına göre daha yoğundur. İş uygulamaları, farklı zaman birimlerinde oluşan iş yoğunlukların başarı ve performans ile çalışabilmesi için elastik çalışma ortamlarına ihtiyaç duymaktadır.

Şirketler, iş süreçlerini devamlı, en az maliyet ve başarılı olarak gerçekleştirmek amacı ile uygulamalarını **Cloud Computing** sağlayıcı şirketler ve özellikle **Microsoft Windows Azure Platform**'a taşımaktadır.

İş uygulamalarının **Windows Azure Platform**'a taşınması ile uygulamaların, kullanabileceği zengin alt yapı sağlanmaktadır. Uygulamaların çalışma ortamın'da **Windows Azure Platform** gibi zengin bir ortama taşınması, kullanıcıları tarafında meydana gelecek olan iş yükünün düzenlenmesine ve uygulama üzerinde bulunan stresin azaltılmasını sağlayacaktır. Ama **Windows Azure Platform** 'un sağlamış olduğu alt yapı gücün amaçlandığı gibi kullanılmasını sağlamayacaktır. İş uygulamaları, **Windows Azure Platform** içerisinde en iyi başarı ve performansı yakalayabilmesi için, uygulamaların **Windows Azure Platform** 'a uyumlu olarak geliştirilmesi gerekmektedir.

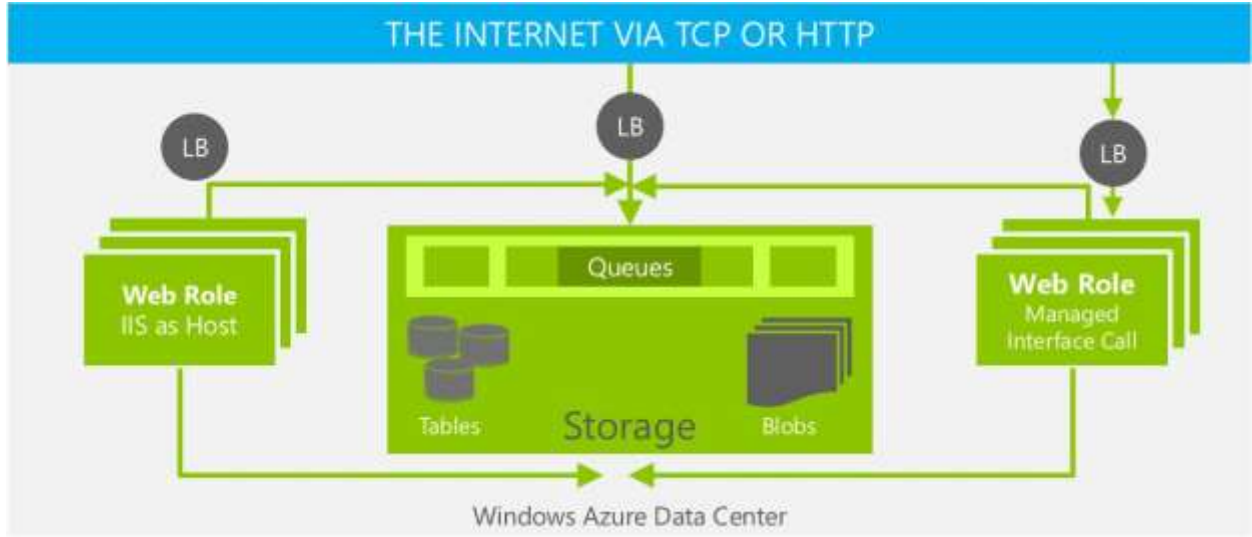
Uygulama yaşam ortamları, uygulamalar üzerinde bulunan iş yoğunluğu düzenlemesini sağlamaktadır. Kullanılan uygulamaların yazılım mimari yapısı ise, uygulamanın sağlık ve başarılı çalışmasını sağlamaktadır.

Geliştirilen iş uygulamalarının, **Windows Azure Platform** alt yapı zenginliklerini en iyi şekilde kullanılabilmesi için **Windows Azure Platform** 'un desteklemiş olduğu, **Windows Azure Cloud Service** uygulama modelinin kullanılması önerilmektedir. Uygulamalar, **Windows Azure Cloud Service** modeli ile kolay yönetilebilir, güvenli ve elastiki yaşam ortamı kazanmaktadır.

Uygulama geliştiricilerinin, Windows Azure Platform zenginleri ve Windows Azure Cloud Service uygulama modelinin hızlı ve kolay öğrenebilmesi amacı ile "WindowsAzure.FunnyApp" uygulama örneği hazırlanmıştır. Uygulama örneği ile temsili olarak iş yoğunlukların dağıtılması ve katmanlı çalışma süreçleri incelenmiştir. Uygulama örneğinde aşağıda belirtilen, senaryolar gerçekleştirilmiştir.

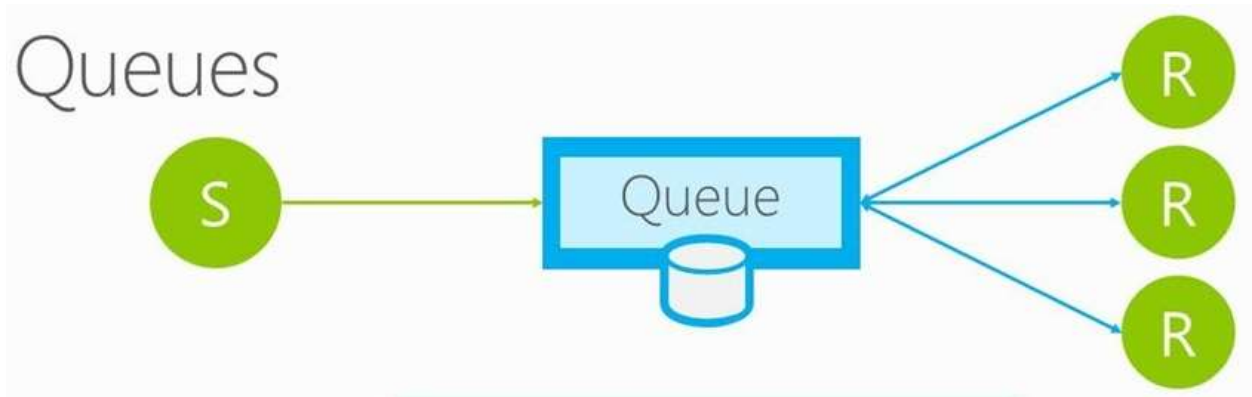
- Kullanıcı hesabının işlemleri (kullanıcı adı, parola, eposta)
- Resim yükleme alanının oluşturulması (istenen resmin yüklenmesi, açıklama, ilgili etiketler)
- Yüklenen resimlerin thumbnail boyutlarında şekillendirilmesi (Windows Azure Worker Role)
- Yüklenen resimlerin görüntülenmesi (thumbnail boyutunda resimlerin listelenmesi)
- Resminde detay gösterilmesi (açıklama, etiketler ve yüklene boyutlarda resim)
- Yorum giriş alanının oluşturulması (kullanıcı adı, eposta ve yorum)
- Yorumların listelenmesi (Kullanıcı adı ve yorum gösterilmesi)

İş uygulamaları içerisinde kullanıcı işlemlerinin gerçekleştirildiği alanlar olduğu gibi yoğun iş yükünün olduğu parçalar da bulunmaktadır. İş uygulamalarında örneğin; resim ya da video işleme, yoğun muhasebe ve arama işlemleri gerçekleştirilebilmektedir. Söz konusu işlemler, **Windows Service** uygulamaları ya da kullanıcı işlemlerinin devam ettirebileceği şekilde planlanmaktadır.



Uygulama örneği olan, “**WindowsAzure.FunnyApp**” çalışması içerisinde kullanıcı tarafında yüklenen resim içeriğin, içerik bilgisi alınarak işleme katmanına taşınmaktadır. Yapılan işlem ile kullanıcı çalışmalarına devam ederken, resim işleme süreçleri farklı katmanda tamamlanmaktadır. İşlem sürecinde **Windows Azure** nesneleri kullanarak, katmanlar arasında mesajlaşma işlemleri ile gerçekleştirilmektedir.

Windows Azure Platform içerisinde birbirinden bağımsız olan, uygulama katmanları iletişiminin sağlanması amacı ile **Queues** nesnelerinin kullanılması önerilmektedir. **Queues** nesneleri, **Windows Azure Storage** içerisinde bulunduğu gibi **Windows Azure ServiceBus Queues** yaklaşımı ile de kullanılmaktadır.



Windows Azure Platform içerisinde aynı uygulama domaininde çalışan, katmanlar arası iletişimin sağlanması amacı ile **Windows Azure Storage** içerisinde bulunan **Queues** nesnesi kullanılması önerilmektedir. Uygulama parçalarının dışı ya da iç sistemler ile iletişim sağlaması için **Windows Azure ServiceBus Queues** yaklaşımın kullanılması önerilmektedir. **Windows Azure Platform** 'un desteklemiş olduğu **Queues** nesneleri ile ilgili bilgi ve karşılaştırma aşağıda bulunmaktadır.

Özellik	Windows Azure Queues	Windows Azure Service Bus Queues
Toplu mesaj gönderme	Desteklemiyor	Destekliyor
Toplu mesaj alma	Destekliyor	Destekliyor
WCF Entegrasyonu	Desteklemiyor	Destekliyor
WF Entegrasyonu	Geliştirme Gerekli	Destekliyor
Server-Side Transaction Log	Destekliyor	Desteklemiyor
Zamanlanmış Gönderim	Destekliyor	Destekliyor
En az mesaj boyutu	64KB	256KB
En fazla mesaj boyutu	100TB	1,2,3,4 ya da 5GB
Kuyruk Algoritması		First-In-First-Out (FIFO)
Kullanılan iletişim protokollü	HTTP/HTTPS üzerinde Rest	HTTPS üzerinde Rest
Peek fonksiyon desteği	Destekliyor	Desteklemiyor
Authentication	Symmetric key	ACS claims
Yetki tabanlı iletişim	Desteklemiyor	Destekliyor

Gerçekleştirilen uygulama örneği, dış herhangi sistem ile iletişim kurmaması nedeni ile **Windows Azure Queues** kullanılmıştır. Uygulama içerisinde mesaj içeriğinin gönderilmesi ve dinleyici tarafından alınması ile ilgili süreçler gerçekleştirilmiştir. Uygulama senaryoları, aşağıda kaynak kodlar ile incelenmektedir.

Aşağıda anlatılan süreçler de kullanılmak üzere bazı sabitler tanımlanmıştır. Tanımlanan değişkenler, yazılan kod içerisinde anahtar içeriklerin tek noktada kontrolünü sağlamaktadır. **Blob, Queue** nesneleri için tanımlanan değişken değerlerinin, küçük harf ile yazılmasına dikkat edilmelidir.

```
public class Utils
{
    public const string ConfigurationString = "DataConnectionString";

    public const string CloudQueueKey = "imagequeue";
    public const string CloudBlobKey = "imageblob";
}
```

Uygulama Senaryosu – I

Kullanıcı tarafında resim ve ilgili bilgilerin girilmesi işlemleri gerçekleştirilmektedir. Gerçekleştirilen işlemler ile ilgili bilgiler mesaj kuyruğuna girmesini sağlayacaktır. Konu ile ilgili kaynak kodlar, yorumlar ile aşağıda anlatılmıştır.

```
private static readonly object _lock = new object();
private static bool _storageInitialized = false;

// işlem gerçekleştirecek olan nesnelerin tanımlanması
```

```

private static CloudBlobClient _blobClient;
private static CloudQueueClient _queueClient;

protected void Page_Load(object sender, EventArgs e)
{
    this.Page.Title = "Image Uploads";
    if (IsPostBack) return;
    InitializeStorage();
}

protected void ButtonSave_Click(object sender, EventArgs e)
{
    if (FileUploadImage.HasFiles & Page.IsValid)
    {
        string uniqueBlobName = string.Format("{0}/funnyimage_{1}{2}", Utils.CloudBlobKey,
                                                Guid.NewGuid().ToString(),
                                                Path.GetExtension(FileUploadImage.FileName));

        CloudBlockBlob blob = _blobClient.GetBlockBlobReference(uniqueBlobName);
        blob.Properties.ContentType = FileUploadImage.PostedFile.ContentType;
        blob.UploadFromStream(FileUploadImage.FileContent);

        FunnyAppRepository<Post> postRepository = new FunnyAppRepository<Post>();
        FunnyAppRepository<Tag> tagRepository = new FunnyAppRepository<Tag>();

        MembershipUser user = Membership.GetUser(Page.User.Identity.Name);
        if (user != null)
        {
            Post post = new Post
            {
                PostContent = TextBoxDescription.Text,
                PostTitle = TextBoxTitle.Text,
                State = false,
                UserId = user.ProviderUserKey.ToString()
            };

            string[] tags = TextBoxTag.Text.Split(';');
            foreach (string tag in tags)
            {
                if (!string.IsNullOrEmpty(tag))
                {
                    tagRepository.Create(new Tag()
                    {
                        PostRowKey = post.RowKey,
                        PostPartitionKey = post.PartitionKey,
                        TagName = tag,
                    });
                    tagRepository.SubmitChange();
                }
            }

            postRepository.Create(post);
            postRepository.SubmitChange();
        }
    }
}

```

```

        // Kuyruk nesneleri
        CloudQueue queue = _queueClient.GetQueueReference(Utils.CloudQueueKey);
        // mesaj içeriğinin oluşturulması
        // mesaj içerisinden birden fazla bilgi olması sebebi ile "," karakteri
        ile bilgiler birbirinden ayrılmıştır.
        CloudQueueMessage message =
            new CloudQueueMessage(string.Format("{0},{1},{2}", blob.Uri, post.P
artitionKey, post.RowKey));
        // Mesaj kuyruğa eklenmiştir.
        queue.AddMessage(message);

        LabelResult.Text = "Uploaded";
    }
    else
    {
        LabelResult.Text = "Failed";
    }
}

private void InitializeStorage()
{
    if (_storageInitialized)
    {
        return;
    }

    lock (_look)
    {
        if (_storageInitialized)
        {
            return;
        }

        try
        {
            // hesap bilgilerinin alınması
            CloudStorageAccount storageAccount = CloudStorageAccount.FromConfigurat
ionSetting(Utils.ConfigurationString);

            // image blob taşıyıcısının oluşturulması
            _blobClient = storageAccount.CreateCloudBlobClient();
            CloudBlobContainer container = _blobClient.GetContainerReference(Utils.
CloudBlobKey);
            container.CreateIfNotExist();

            // Blob taşıyıcısına ile ilgili erişim ayarlarının tanımlanması
            var permissions = container.GetPermissions();
            permissions.PublicAccess = BlobContainerPublicAccessType.Container;
            container.SetPermissions(permissions);

            // create queue to communicate with worker role
            _queueClient = storageAccount.CreateCloudQueueClient();
            CloudQueue queue = _queueClient.GetQueueReference(Utils.CloudQueueKey);

```

```

        queue.CreateIfNotExist();
    }
    catch (WebException exception)
    {
        Trace.Write(exception.Message);
    }

    _storageInitialized = true;
}
}

```

Uygulama Senaryosu – II

Mesaj kuyruğunun dinlenmesi ve ilgili bilgilerin alınması gerekmektedir. Söz konusu bilgilerin alınması ile **Worker Role** içerisinde resim işleme süreci istenilen boyutlara şekillendirilecektir. Dinleme işlemi, uygulama yaşam süresi boyunca, kullanıcı taleplerinin gerçekleştirilmesi amacı ile sonsuz bir döngü içerisinde gerçekleştirilmektedir.

```

// işlem gerçekleştirecek olan nesnelerin tanımlanması
private CloudQueue _queue;
private CloudBlobContainer _container;

public override void Run()
{
    // Worker nesnesi içerisinde sürecin sürekli olarak gerçekleştirilmesi amacı il
    e
    // sonsuz döngüye alınması gerekmektedir.
    while (true)
    {
        try
        {
            // İşlem sürecinde kuyruk içerisinde bulunan mesajın dinleme işlemi
            CloudQueueMessage message = _queue.GetMessage();
            if (message != null)
            {
                string[] messageArray = message.AsString.Split(new char[] { ',' });
                string outputBlobUri = messageArray[0];
                string partitionKey = messageArray[1];
                string rowkey = messageArray[2];

                // Kuyruk içerisinde okunan mesajı, döngü içerisinde tekrar okuması
                önemek amacı ile
                // mesaj Peek edilmektedir.
                _queue.PeekMessage();

                string inputBlobUri = Regex.Replace(outputBlobUri, "([^\.\.]+)(\\.\.[^
                \\.]+)?$", "$1-myimage$2");

                _container.CreateIfNotExist();
                CloudBlob inputBlob = _container.GetBlobReference(outputBlobUri);
                CloudBlob outputBlob = _container.GetBlobReference(inputBlobUri);

                var u = inputBlob.Uri.AbsolutePath;
            }
        }
        catch { }
    }
}

```

```

        // Blob nesnesi içerisinde bulunan resim içeriği okunması
        using (BlobStream input = inputBlob.OpenRead())
        using (BlobStream output = outputBlob.OpenWrite())
        {
            ProcessImage(input, output);

            output.Commit();
            outputBlob.Properties.ContentType = "image/jpeg";
            outputBlob.SetProperties();

            FunnyAppRepository<Post> postRepository = new FunnyAppRepository
y<Post>();

            Post post = postRepository.Find(partitionKey, rowkey);

            post.PostImage = inputBlobUri;
            post.State = true;
            postRepository.Update(post);
            postRepository.SubmitChange();

            _queue.DeleteMessage(message);
        }
    }
}
catch (StorageClientException e)
{
    Trace.Write(e);
}
}

public override bool OnStart()
{
    // Worker üzerinde başlatılması amaçlanan varsayılan bağlantı sayısı
    ServicePointManager.DefaultConnectionLimit = 12;
    // Uygulama içerisinde bağlantı bilgilerinin alınma işlemi
    CloudStorageAccount.SetConfigurationSettingPublisher((configName, configSetter)
=>
        configSetter(RoleEnvironment.GetConfigurationSettingValue(configName)));

    // Uygulama içerisinde bağlantı bilgilerinin alınma işlemi
    var storageAccount = CloudStorageAccount.FromConfigurationSetting(Utils.Configu
rationString);

    // kuyruk nesnesi ile ilgili refcasnların oluşturulması
    CloudQueueClient queueStorage = storageAccount.CreateCloudQueueClient();
    _queue = queueStorage.GetQueueReference(Utils.CloudQueueKey);

    // kuyruk nesnesinin bulunup-bulunmadığı kontrollünün yapılması
    _queue.CreateIfNotExist();

    CloudBlobClient blobStorage = storageAccount.CreateCloudBlobClient();
    _container = blobStorage.GetContainerReference(Utils.CloudBlobKey);

    _container.CreateIfNotExist();
}

```

```
        return base.OnStart();
    }

    // Resim boyutlandırma işlemleri ile fonksiyon
    public void ProcessImage(Stream input, Stream output){ ... }
```

İş uygulamalarının, her yeni gün daha fazla kullanılması ile iş yükleri artırılmaktadır. Gerçekleştirilen her sürecin, devamlı ve başarılı sonuçlar üretmesi beklenmektedir. Süreçlerin planlandığı gibi gerçekleşmesi, süreç içerisinde kullanılan uygulamanın sağlıklı çalışması büyük önem taşımaktadır.

Not: Yapılan anlatımın örneklenmesi amacı ile “**WindowsAzure.FunnyApp**” uygulaması hazırlanmıştır. Aşağıdaki bağlantı kullanarak, uygulama kaynak kodlarına erişebilirsiniz.

Github / <https://github.com/ibrahimatay/WindowsAzure.FunnyApp>

İş uygulamaların başarılı ve performanslı çalışması, uygulama yazılım mimarisi ve konumlandırıldığı platform ile ilişkili olarak gerçekleşmektedir. **Windows Azure Cloud Service** uygulama geliştirme modeli ile geliştirilen uygulamalar, iş sürecine bağlı olarak parçalarına ayrılarak, beklenen başarı ve performans elde edilebilmektedir.

EK Not: Bölüm ile ilgili uygulama videosu ve sunumunu incelemek için aşağıdaki bağlantıyı kullanabilirsiniz.

[http://www.ibrahimatay.org/post/2013/4/18/Windows-Azure-ile-Cloud-Computing-Uygulamalari%E2%80%94\(-Video-\).aspx](http://www.ibrahimatay.org/post/2013/4/18/Windows-Azure-ile-Cloud-Computing-Uygulamalari%E2%80%94(-Video-).aspx)

Zaman içerisinde planlı ya da plansız olarak, birçok işlem yapılmaktadır. Günümüzde sosyal ağlar ile paylaşımlar ya da bankalar ile birçok harcamalar yapmaktayız. Yapılan işlemler kullanıcılar için önemli olmasa da şirketler için CRM bilgisi olarak depolanmaktadır.

Günümüzde müşteri isteklerini, talep etmeden hazırlayan şirketler kazanmaktadır.

Yeni nesil CRM süreçleri, her yeni gün özellik kazanarak şekillenmektedir. Söz konusu CRM süreçleri, yasal yaptırımlara bağlı gerçekleştirilmektedir.

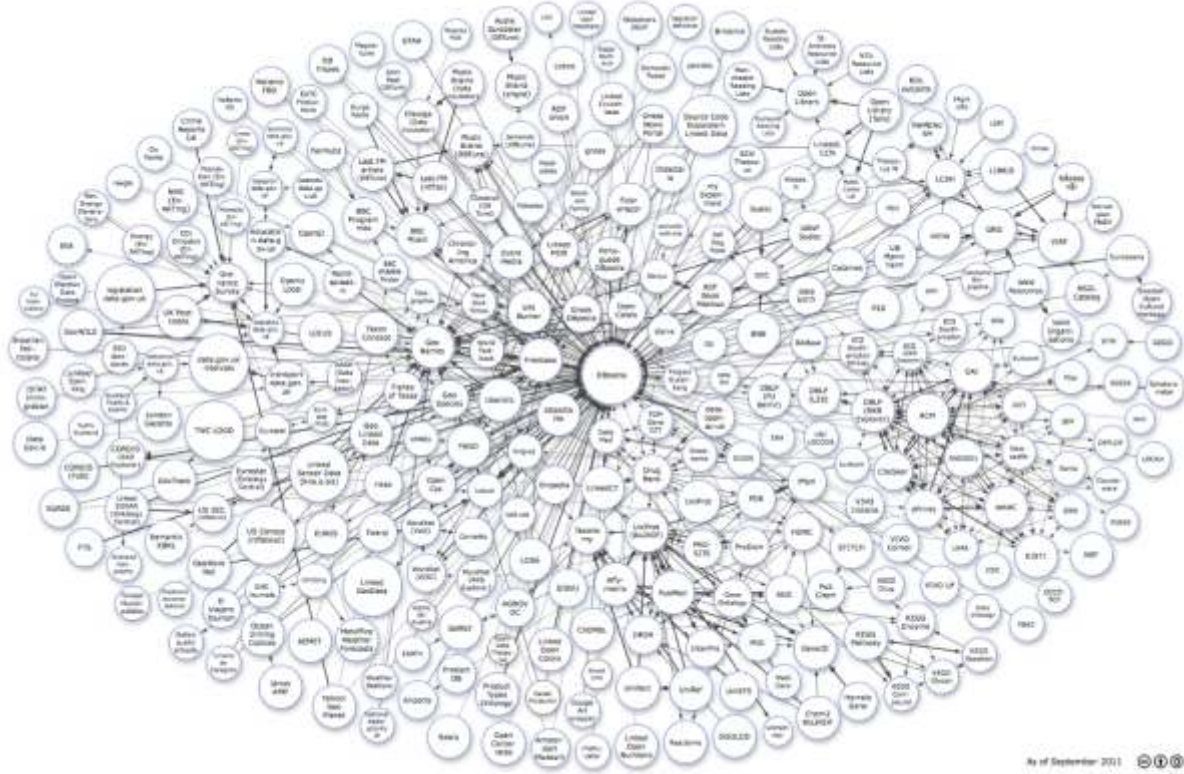


İş planlama ve senaryoları gereksinim ve yasal yaptırımlara uyumluluk sağlanmak amacı ile güncellenmektedir. Uygulamaların kullanım sürecinde gereksinim duyulmayan, herhangi özellik dönemde gereksinim duyulabilmektedir. Gerçekleşen değişimler, kullanılan iş uygulamalarının şekillendirilmesi ya da yeniden geliştirilmesi gündeme gelebilmektedir.

Türkiye Cumhuriyeti'nde, 1999 yılın sonu itibari ile vatandaşlık numarası uygulaması başlatılmıştır. Uygulama ile isim benzerliklerinde kaynaklanan problemler ve tüm işlemlerin tek numara üzerinde gerçekleştirilmesi amaçlanmıştır. Çalışma ile başta sağlık sektörü olmak üzere, insan ile ilişkili tüm sektörlerde kullanılan uygulamalarda, vatandaşlık numarasını temel alma özelliğinin kazandırılması gereksinimi ortaya çıkmıştır.

Yapılan işlemler ile tamamlanmış mevcut ya da süren işlem verilerinin düzenlenmesi gereksinimi ortaya çıkmıştır.

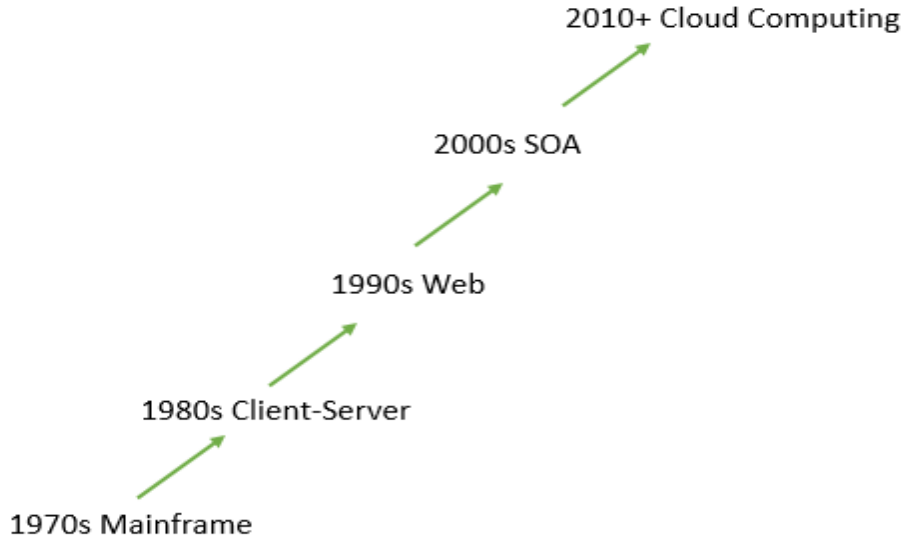
İş süreçleri, kural ya da yasal yaptırımların uygulanması sebebi ile işlem girdi ve çıktıların değişiklik göstermesi gündeme gelebilmektedir. Yapılan işlemler, mevcutta verilerin düzenlenmesi, performans ve depolama süreçlerini de etkilemektedir.



İş uygulamaları, kullanıcılarından yazılı ya da Binary türünde çeşitli veriler ile çalışmaktadır. Çalışılan veriler, yığınların oluşmasına neden olmaktadır. Veri yığınları ise, zaman ile büyük veri (**Big Data**) olgusunu gündeme getirmektedir.

Her şirketin çalışmış olduğu ana iş dalı üzerinde çalışması, konu ile ilgili uzmanlaşmasını sağlayacaktır. Gereksinimleri duyduğu diğer yan ihtiyaçları, konunun uzmanlarında sağlaması, iş sürecini ve üretimin kalitesini artıracaktır.

Şirketler, mevcut veri yığınlarını, depolama, analiz ve yönetim için çeşitli iş uygulamaları kullanmaktadır. Veri yığınları ile ilgili işlem yapılabilmek amacı ile çeşitli araçlar satın alınmaktadır. Yapılan satın alma hareketleri, her şirketin birer veri merkezinin oluşturmasına neden olmaktadır. Veri merkezlerinin kurulması, yönetilmesi, bakımı ya da nitelikli iş gücünün oluşturulması, şirketlerin iş kolları dışında yüklerin altına girmesine neden olmaktadır.



Günümüzde sektör ve iş kolların çeşitlenmesi, iş ihtiyaçlarının ile ilgili duyulan verinin artmasına neden olmaktadır. İş gereksinimleri sağlanması ve istenilen sonuçların üretilmesi için iş uygulamalarının, **Cloud Computing** sayılayıcılara ve özellikle **Microsoft Windows Azure Platform** 'a taşınmaktadır.

İş uygulamaları, gerçekleştirilmesi istenen süreci en avantajlı sunması amacı ile çeşitli teknolojik bağımlıklara yüklenmektedir. Çeşitli nedenlerden dolayı, değişen iş süreçleri, kullanılan uygulamaların da güncelleme gereksinimini ortaya çıkartmaktadır. İş Uygulamaları, iş süreçleri ile ilgili çeşitli avantajlar sunarken, değişim süreçlerinde handikapların oluşmasına neden olabilmektedir.

Uygulama mimarileri, iş süreç ve yasal bağımlıkları nedeni ile kullanıcı tarafından verilerin güncellenmesi gündeme gelebilmektedir. Uygulama veri yapılarının, zaman ve ihtiyaçların gereğince değişmesi, uygulama performansı ve veritabanı optimizasyonu gibi problem oluşturabilmektedir. Geçmiş dönemde gereksinim olduğu düşünülen veriler, farklı bir dönemde gereksinim duyulmayabilir. Benzer yaklaşım olarak, bazı uygulama verileri gereksinim duyulabilmektedir. Yapılan işlemler, verinin depolanması ya da analiz sürecini etkileyebilmektedir.

No	Name	SurName	Status	Title	BirthDate	Phone
1	Davolio	Nancy	FALSE	*	*	*
2	Fuller	Andrew	FALSE	*	*	*
3	Leverling	Janet	FALSE	Sales Representative	*	(206) 555-8122
4	Peacock	Margaret	FALSE	Sales Representative	*	(206) 555-1189
5	Buchanan	Steven	TRUE	Sales Manager	4/3/1955	*
6	Suyama	Michael	TRUE	Sales Representative	2/7/1963	*

Gelişen teknoloji ile iş süreçlere uygun, kolay ve hızlı iş çözümleri oluşturulabilmektedir. Günümüz iş uygulamalarının, hızlı veri sorgulama ve veri ile ilgili yapısal düzenleme

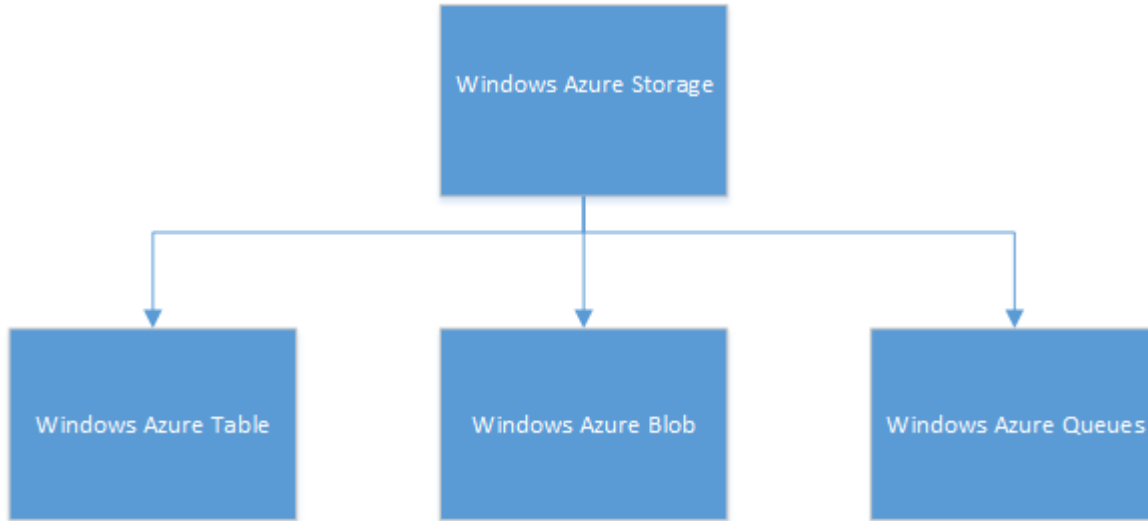
gereksinimleri ortaya çıkarmıştır. Meydana gelen gereksinimler, verinin yapının esnek olması ihtiyacını doğurdu.

2009 yılın da "eğlenceyi seç, ilişkisel=yanlış; olan gerçek Dünyadan faydalan" sloganı ile yapılan, "no:sql(east)" konferansı ile **NoSQL** konusunda çeşitli düşüncelerin oluşmasında büyük etki oluşturmuştur.

NoSQL, 1998'un sonlarına doğru ortaya çıkan bir kavramdır. Klasik ilişkisel veritabanı yapışanda bulunmayan ve sorgulama için SQL dili kullanmayan veritabanı türüdür. Genel olarak xml ya da JSON formatında veri depolama yapmaktadır.

İş ihtiyaçları, iş verisinin esnek olma gereksinimi beraberinde de ihtiyacın hizmet olarak sunulmasını da sağlamıştır. **Microsoft** ve diğer **Cloud Computing** sağlayıcıları çeşitli depolama hizmetleri ile kullanıcılarına **Cloud Storage** hizmetleri sunmaktadır. **Microsoft** da **Windows Azure Platform** ile kullanıcılarına ilişkisel(**SQL Database**) veritabanı hizmeti sunduğu gibi No-Relation(**Windows Azure Storage**) iş çözümü de sunmaktadır.

Windows Azure Platform No-Relation iş çözümü olarak, **Windows Azure Storage** hizmeti sağlamaktadır. Hizmet ile Binary ve diğer veri türleri farklı nesneler içerisinde güvenli ve yüksek optimizasyonu değerlerinde, depolanması sağlanmaktadır. Aşağıda **Windows Azure Storage** hizmeti katmanları belirtilmiştir.



Windows Azure Storage hizmeti, üç parçalı ve parçalara özgü özelliklerin olması sebebi ile anlatımın devamında **Windows Azure Storage** 'ın parçalarından olan **Windows Azure Table Storage** ile ilgili bilgiler verilecektir.

Windows Azure Table Storage, **SQL Database** yaklaşımında bulunan **Table** nesnesine benzemektedir. Ama yapısal bazı özgü özelliklerden dolayı, **Windows Azure Table Storage** yapısı **SQL Database**'de bazı farklıklar göstermektedir. Aşağıda **Windows Azure Storage** ve **SQL Database** ile ilgili bilgi ve karşılaştırmalar bulunmaktadır.

Özellik	Windows Azure Table Storage	SQL Database
İlişkisel veri	Desteklemiyor	Destekliyor
Server-side çalışma	Desteklemiyor	Destekliyor
Geo-replication	Destekliyor	Desteklemiyor
Scale-out	Otomatik	Manuel
LINQ desteği	Destekliyor	Destekliyor
Veri erişimi	OData Protokol	ODBC ya da JDBC
Yönetim protokollü	HTTP/HTTPS üzerinde REST	HTTP/HTTPS üzerinde REST ile ODBC/JDBC
En az veri depolama	1MB	2GB
En fazla veri depolama	100TB Tablo başına	150GB Tablo başına
Firewall Güvenliği	Desteklemiyor	Destekliyor
REST protokol ile erişim	Destekliyor	Destekliyor
Transaction	Destekliyor (Limitli)	Destekliyor
Transaction logs tutma	Desteklemiyor	Destekliyor

Windows Azure Table ile çalışmak, veritabanı işlemlerini **Entity Framework** ile yapmak kadar kolay, işlevsel ve hızlıdır. **Windows Azure Table Storage**, veri işlem süreçlerin de **LINQ** sorgularını kullanılabilir. **Entity Framework** ile geliştirilen uygulamalarda **DbContext** nesnesi kullanıldığı gibi **Windows Azure Table Storage** yapısında ise, **TableServiceContext** nesnesi kullanılmaktadır.

```
public class FunnyAppContext : TableServiceContext
{
    public FunnyAppContext(string baseAddress, StorageCredentials credentials)
        : base(baseAddress, credentials)
    {
    }

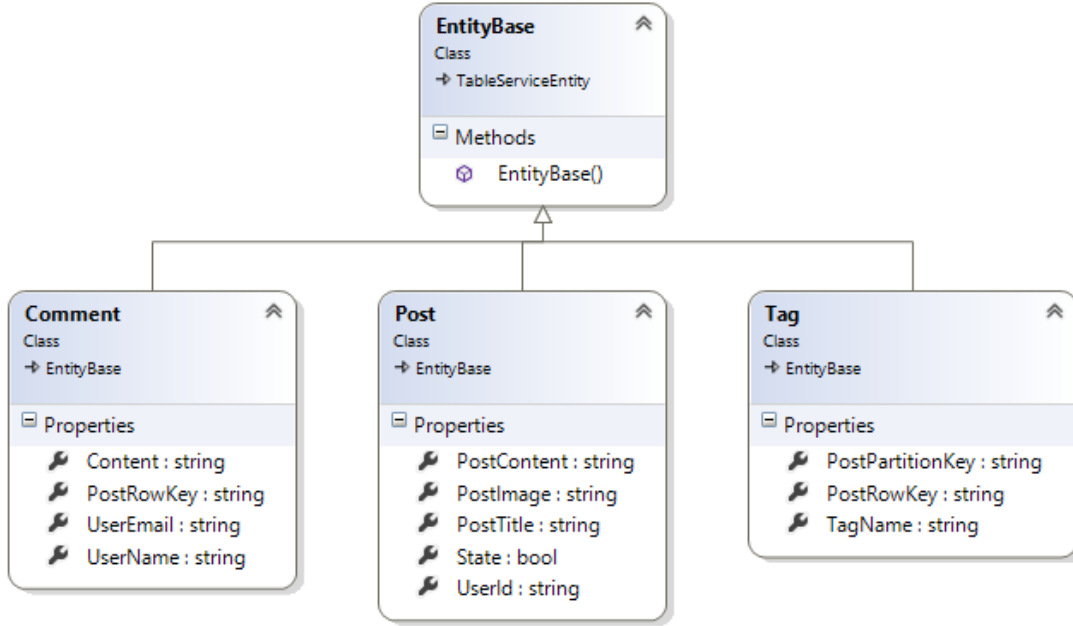
    public IQueryable<Tag> Tag
    {
        get { return this.CreateQuery<Tag>("Tag"); }
    }

    public IQueryable<Post> Post
    {
        get { return this.CreateQuery<Post>("Post"); }
    }

    public IQueryable<Comment> Comment
    {
        get { return this.CreateQuery<Comment>("Comment"); }
    }
}
```

```
}
```

Uygulama örneği olan, “WindowsAzure.FunnyApp” çalışması nesneye yönelimli olarak geliştirilerek, gerçek yaşamda karşılaşılabilecek durumlar ele alınmaya çalışılmıştır. Aşağıda “WindowsAzure.FunnyApp” çalışması ile ilgili sınıf ilişkisel şeması bulunmaktadır.



Yukarıdaki şema da görüldüğü gibi **EntityBase** sınıfı **TableServiceEntity** sınıfında türetilmiştir. Yapılan türetme ile **Windows Azure Table Storage** Entity fonksiyonelliği kazandırılmış olacaktır. Temel sınıf olan **EntityBase** sınıfından türeyen sınıflar, “**PartitionKey**”, “**RowKey**” ve “**Timestamp**” özellikleri kazanmaktadır. Söz konusu özellikler, **Windows Azure Table Storage** içerisinde verilerin, benzersiniz(“**PartitionKey**”, “**RowKey**”) ve işlem zamanı(“**Timestamp**”) ile ilgili bilgiler taşımaktadır.

```
public class EntityBase : TableServiceEntity
{
    public EntityBase()
    {
        this.PartitionKey = DateTime.UtcNow.ToString("MMddyyyy");
        this.RowKey = string.Format("{0:10}_{1}", DateTime.MaxValue.Ticks - DateTime.Now.Ticks, Guid.NewGuid());
    }
}
```

Klasik uygulama veritabanı uygulama geliştirme sürecinde gerçekleştirildiği gibi **Windows Azure Table Storage** ile de GRUD işlemleri yapılabilmektedir. Yapılan işlemlerin bir kalıp üzerinde gidebilmesi amacı ile Repository tasarım deseni kullanılmıştır.

```
public interface IRepository<TEntity>
{
    TEntity Find(string partitionKey, string rowKey);
    TEntity Find(string rowKey);
    void Create(TEntity entity);
    void Delete(TEntity entity);
    void Update(TEntity entityToUpdate);
    void SubmitChange();
    IQueryable<TEntity> Get();
}
```

Windows Azure Table Storage nesnesi ile veri işlemlerinin yönetebilmesi amacı için Repository nesnesinin yönetilebildiği FunnyAppRepository sınıfı hazırlanmıştır. Sınıf **EntityBase** türüne ait nesnelere hizmet vermek amacı ile sınırlandırılmıştır.

```
public class FunnyAppRepository<TEntity> : IRepository<TEntity>,
    IDisposable where TEntity : EntityBase
{
    // veri işlemleri ile ilgili hesap
    // bilgilerinin alınması
    private static CloudStorageAccount _storageAccount;

    // iş nesnesi ile ilgili tanımlamanın yapılması
    private readonly FunnyAppContext _context;

    // nesne türüne ayit isim alınması
    private readonly string _entitySetName;

    public FunnyAppRepository()
    {
        // veri işlemleri ile ilgili gerekli bilgilerin alınması
        _storageAccount =
            CloudStorageAccount.FromConfigurationSetting(Utils.ConfigurationString)
;

        // veri hesabı bilgilerine bağlı olarak,
        // veri nesnelerinin Windows Azure Storage üzerinde oluşturulması
        CloudTableClient.CreateTablesFromModel(
            typeof (FunnyAppContext),
            _storageAccount.TableEndpoint.AbsoluteUri,
            _storageAccount.Credentials);

        // seçili olan nesne türünün isminin alınması
        _entitySetName = typeof (TEntity).Name;

        // seçili olan nesne ile ilgili gerekli tanımlamaların
        // Windows Azure Storage içerisinde olup/olmadığını kontrollünün yapılması.
        // Söz konusu nesnenin Windows Azure Storage içerisinde olmaması durumunda
        // nesne ismine bağlı olarak, gerekli tanımlamaları yapılması
    }
}
```

```

        _storageAccount.CreateCloudTableClient().CreateTableIfNotExist(_entitySetName);

        // GRUD işlemlerinde sorumlu olan, iş nesnesine erişim ile ilgili
        // gerekli bilgilerin atanması
        this._context = new FunnyAppContext(_storageAccount.TableEndpoint.AbsoluteURI,
        _storageAccount.Credentials);
        this._context.RetryPolicy = RetryPolicies.Retry(3, TimeSpan.FromSeconds(1))
        ;

        }

        // nesne türüne bağlı olarak
        // rowkey ve partitionkey ile kayıt getirme
        public TEntity Find(string partitionKey, string rowKey)
        {
            return (from g in _context.CreateQuery<TEntity>(_entitySetName)
                    where g.PartitionKey == partitionKey && g.RowKey == rowKey
                    select g).FirstOrDefault();
        }

        // nesne türüne bağlı olarak
        // rowkey ile kayıt getirme
        public TEntity Find(string rowKey)
        {
            return (from g in _context.CreateQuery<TEntity>(_entitySetName)
                    where g.RowKey == rowKey
                    select g).FirstOrDefault();
        }

        // nesne türüne ait yeni üye oluşturma
        public void Create(TEntity entity)
        {
            this._context.AddObject(_entitySetName, entity);
        }

        // nesne silmek için
        public void Delete(TEntity entity)
        {
            this._context.DeleteObject(entity);
        }

        // nesne güncellemek için
        public void Update(TEntity entityToUpdate)
        {
            this._context.UpdateObject(entityToUpdate);
        }

        // yapılan değişiklikleri depolama alanına yansıtmak için
        public void SubmitChange()
        {
            this._context.SaveChanges();
        }

        // nesne türüne ait tüm içerikleri çekebilmek için

```

```

public IQueryable<TEntity> Get()
{
    return this._context.CreateQuery<TEntity>(_entitySetName);
}

public void Dispose()
{
    GC.SuppressFinalize(this);
}
}

```

Tanımlaması yapılan iş nesnelerinin, kullanımı ile uygulama senaryosu olan, Kullanıcın tarafında yüklenen fotoğrafların görüntülenme süreci gerçekleştirilmektedir. Yapılan işlem ile **LINQ** sorgusu kullanarak gerçekleştirilmektedir.

```

Protected void Page_Load(object sender, EventArgs e)
{
    if (IsPostBack) return;
    // Post nesne türüne ayıt, repository iş nesnesinin tanımlanması
    FunnyAppRepository<Post> _postRepository = new FunnyAppRepository<Post>();

    List<PostViewData> viewDatas = new List<PostViewData>();

    // istenilen öncüllerin oluşturulduğu Linq sorgusu
    _postRepository.Get().Where(post => post.UserId == Membership.GetUser(Page.User
.Identity.Name)
.ProviderUserKey.ToString()).ToList()
.Where(post=> post.State)
.OrderByDescending(post => post.Timestamp)
.Take(20).ToList().ForEach(post => viewDatas.Add(new PostViewDat
a()
{
    PostContent = post.PostContent,
    PostImage = post.PostImage,
    RowKey = post.RowKey,
    UserId = post.UserId
})));

    this.RepeaterImages.DataSource = viewDatas;
    this.RepeaterImages.DataBind();
}

```

Her yeni gün, yeni çalışmalar ve işlemler yapmaktayız. Gerçekleştirilen süreçler, veri yığınları oluşmasına neden olmaktadır. Meydana gelen veri yığınları, güncelleme, analiz ve depolama gibi gereksinimleri ortaya çıkarmaktadır. Yaşanan problemler, ortaya çıkan **NoSQL** yaklaşımları ile amaçlanmaktadır.

Veri yapılarının hatalı analiz ve planlanması nedeni ile veri ile ilgili tüm alanlarda problemlerin ortaya çıkması mümkün olacaktır.

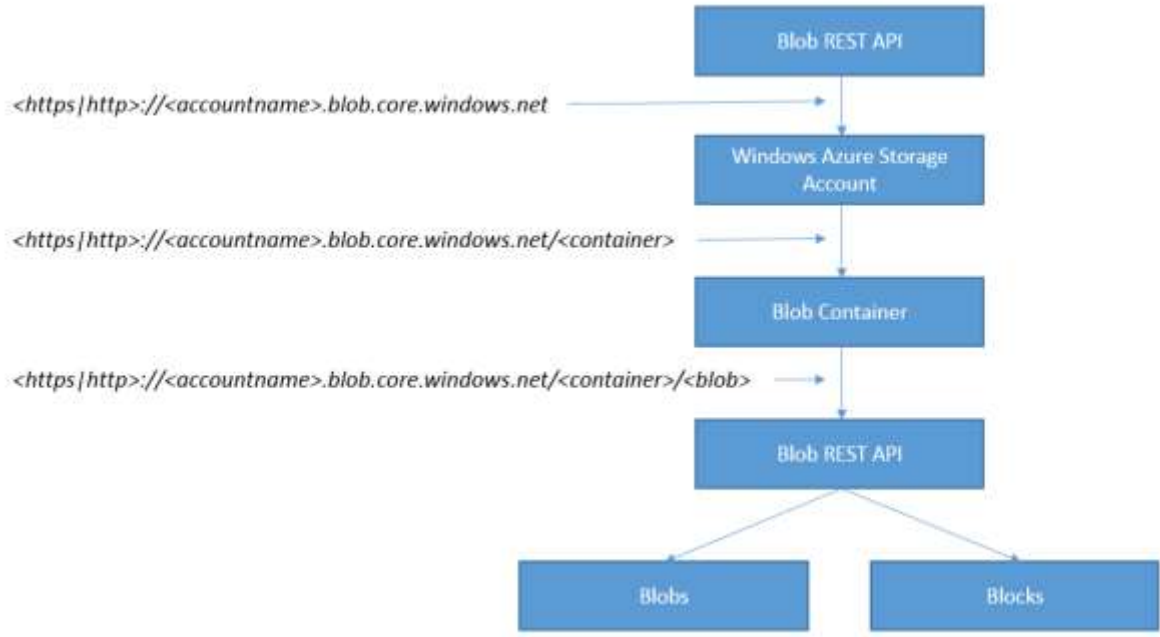
Not: Yapılan anlatımın örneklenmesi amacı ile “**WindowsAzure.FunnyApp**” uygulaması hazırlanmıştır. Aşağıdaki bağlantı kullanılarak, uygulama kaynak kodlarına erişebilirsiniz.

Github / <https://github.com/ibrahimatay/WindowsAzure.FunnyApp>

Windows Azure Storage yaklaşımı, geliştiricilerine kullandıkları verilerin sınıflandırma ve planlamaya yöneltmektedir. Veri yığınlarının, **Windows Azure Storage** uyumluluğunun sağlanması ile esnek ve kolay yönetilebilir yapılara sahip olunmaktadır.

EK Not: Bölüm ile ilgili uygulama videosu ve sunumunu incelemek için aşağıdaki bağlantıyı kullanabilirsiniz.

[http://www.ibrahimatay.org/post/2013/5/22/Windows-Azure-ile-Cloud-Computing-Uygulamalari---7-\(-Video-\).aspx](http://www.ibrahimatay.org/post/2013/5/22/Windows-Azure-ile-Cloud-Computing-Uygulamalari---7-(-Video-).aspx)



Windows Azure ile Cloud Computing Uygulamaları – 8

*İş uygulamaların başarılı ve performanslı olarak çalışabilmesi için geliştiricilerin, iş tecrübelerine göre çeşitli yaklaşımlar uygulanmaktadır. Günümüzde depolama ve verinin kullanımı konusunda yeni konseptler oluşturulmaktadır. Microsoft **Windows Azure Platform** ile de sürece yeni bir bakış açısı oluşturulmuştur.*



Zaman sürekli ilerliyor. Geçmişte büyük prodüksiyonlar ile yapılacağı inanılan çalışmalar, kişisel cihazlar ile gerçekleştirilebilir hale geldi. Yaşanılan anları kayıt etme isteği, herkesi fotoğrafçısı ve rejisörü olabilme yolu açmıştır.

Yaşam hızlı değişiyor. Yaşanan değişimin en büyük mimarı ise teknoloji olarak görünmektedir. Teknoloji kullanıcılarına birçok olanak sağladığı gibi kendisi içerisinde farklı olanaksızlar da bulunmaktadır.



Şirketler, iş süreçlerini standartlar ile devamlılığını sağlayabilmek amacı ile çeşitli teknolojiler kullanılmaktadır. Kullanılan teknolojiler, gerçekleştirilmesi istenen sürecin başarılı ve en az problem ile tamamlaması beklenmektedir. Beklenen sonuçların alınması, müşteri memnuniyetinin sağlanması şirketler için önemli olmaktadır.

İş süreçleri devamlı ve istenilen kalite de yürütülmesi, müşteri ve sağlayıcı(üretici) arasında olumlu ilişkilerin kurulabilmesi için önemli bir bağ olarak görülmektedir. Şirketler sundukları hizmet ya da ürünlerin, müşterilerine tanıtılabilmek amacı ile reklam filmleri ve kataloglar kullanmaktadır. Kullanılan metreyeler, teknoloji sistemlerin kullanılması ile oluşturularak müşterilere ulaştırılmaktadır.

Günümüzde şirketler tanıtımlarını sağlayabilmek amacı ile birçok yöntem kullanmaktadır. Kullanılan yöntemlerin büyük kısmı, teknoloji sistemlerin kullanıldığı ve müşteri memnuniyetin ön planda olduğu sistemlerdir. Örneğin; Atın ve mücevher satışı yapan şirketler, hazırladığı ürünleri Web sitesi üzerinden çeşitli fotoğraflar kullanarak tanıtımını yapmaktadır. Ayrıca her zaman müşterilerinin yanında olabilme olanağı, ürün destek ve satış sürecini için aktif süreçler gerçekleştirebilmektedir. Gerçekleşen süreçler ile şirket, genişleyen müşteri portföyüne sahip olmaktadır.

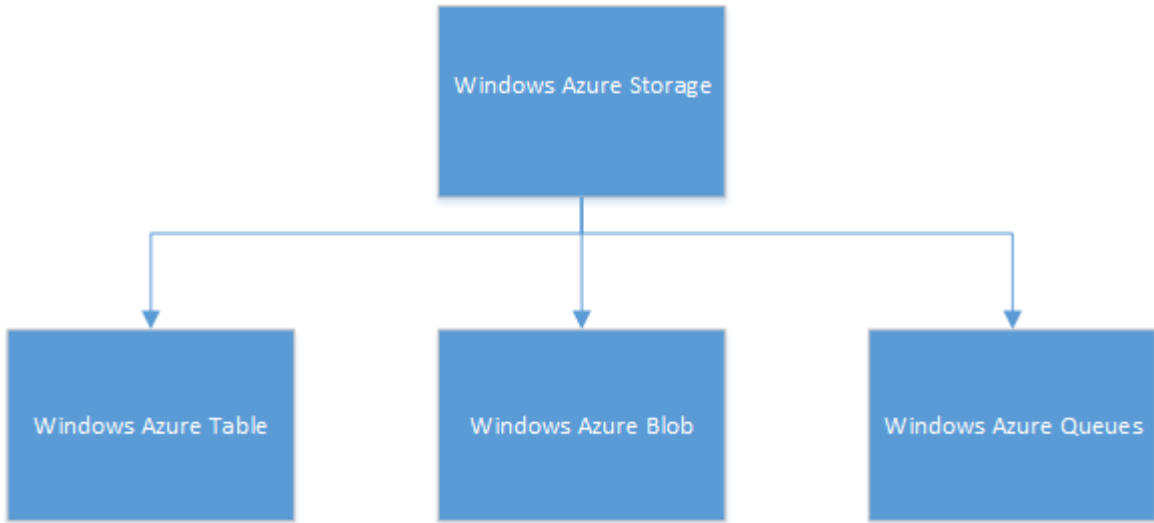
İş dünyasının hızlı manevrana cevap verebilmek, şirketlerin yaşamının devam ettirebilmesi için çok önemlidir. Genişleyen iş süreçleri ile yeni teknoloji **inovasyon**ların önü açılmaktadır. **İnovasyon** hareketleri, geçmişte yapılan hatalarda ders alınarak oluşturulmaktadır. Günümüzün **inovasyon** çözümleri incelendiğinde en belirgin özellikleri sürdürülebilir, ekonomik, esnek ve güvenilir olarak sıralanmaktadır. Belirtilen özellikler **Cloud Computing** mimarisinin en temel özellikleridir.

Cloud Computing mimarisi çeşitli şirketler tarafından yorumlanmaktadır. Özellikle Microsoft, **Windows Azure Platform** ürünü ile kullanıcılarına ekonomik ve kolay yönetilebilir hizmetler sağlamaktadır.

Windows Azure Platform, esnek ve devamlılığı sağlanması istenen birçok iş çözümünün yaşayabilmesi için çeşitli çözümleri sunmaktadır. Sunulan iş çözümleri, yeni birçok konsept kullanarak, iş süreçleri esnek olabilmesine olanak sağlamaktadır.

Şirketler gerçekleştirdiği yoğun iş süreçleri, teknoloji olarak incelendiğinde problemlerin temelinde depolama sorunları ile karşılaşmaktadır. Sürecin güvenli ve en az problem ile sağlanabilmesi amacı ile Microsoft, **Windows Azure Platform** ile yeni depolama konsepti, **Windows Azure Storage** altyapısını sunmuştur.

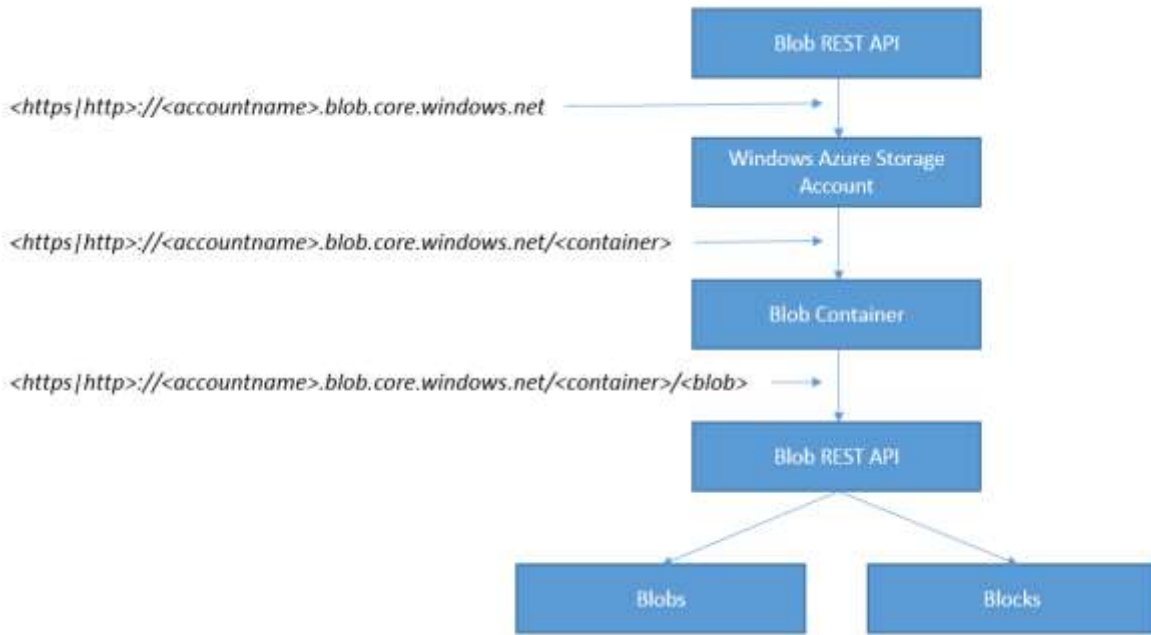
Windows Azure Storage, yüksek optimizasyonu değeri olan ve yüksek güvenliği seviyeli depolama altyapısıdır. Söz konusu altyapı, klasik uygulama geliştirme yaklaşımın ötesinde nesnel ve veri kayıp problemlerin en alt düzeyde olan bir sistemdir. **Windows Azure Storage** altyapısı, iş gereksinimlerine cevap üretecek üç parçadan oluşmaktadır. Aşağıda **Windows Azure Storage** altyapısı parçaları şema ile gösterilmektedir.



Windows Azure Storage, kullanıcı iş gereksinimlerine göre **Table**, **Queue** ve **Blob** isimli üç parçadan oluşmaktadır. **Windows Azure Storage**, **Table** ve **Queue** parçaları ile ilgili "[Windows Azure ile Cloud Computing Uygulamaları](#)" makale serisinin diğer bölümlerinde bilgiler verilmiştir. Bu çalışma ile Windows Azure Blob Storage ile ilgili bilgiler verilmesi amaçlanmaktadır.

İş uygulamaları çoğunlukla kullanıcıların dan çeşitli Binary(fotoğraf ya da Word dosyası) içerikler alınmaktadır. Klasik uygulama geliştirme yaklaşımı incelendiğin de kullanıcıların dan alınan dosya içeriklerinin uygulama makinesi üzerine konumlandırılmaktadır. Kullanıcı dosyaların ve iş uygulaması ile aynı makine üzerine konumlandırılması, zaman içerisinde depolama çeşitli problemlerin ortaya çıkmasına neden olmaktadır.

İş uygulamalarının, **Windows Azure Blob Storage** altyapısı kullanarak geliştirilmesi ile uygulama verilerin depolama yaklaşımı farklı bir konseptte taşınmaktadır. Uygulama verileri, **Windows Azure Storage** ile uygulama alanından bağımsız, esnek ve yüksek güvenli altyapıya taşımaktadır. Aşağıda **Windows Azure Blob Storage** ile ilgili iletişim şeması bulunmaktadır.



Windows Azure Blob Storage, iş gereksinimlerine göre çeşitli olanaklar sunmaktadır. Yukarıda bulunan şema da görüldüğü gibi **Windows Azure Storage**, **REST** mimari ile iletişimini sağlayan, iki ayrı parçadan oluşmaktadır. Aşağıda **Windows Azure Blob Storage** altyapısı ile ilgili bazı özellikler ve açıklamaları bulunmaktadır.

Özellik	Açıklama
Snapshot	Depolanan verinin belirlenen tarihe kadar sadece okunabilir halde dondurulması şeklinde hizmet verme durumudur.
Erişim Seviyesi	Depolanması istenen verinin özel(private), sadece genel(public) okuma ya da genel(public) erişim ile tüm yetkiler sağlanabilmesi şeklinde erişim yetkileri sınırlandırılmaktadır.

Disaster Recovery	Geo-replication özelliği sayesinde depolanan veri bir den fazla lokasyon üzerine kopyalanmaktadır. Gerçekleşen kopyalama ile herhangi noktada yaşanan sistem problemlerin de farklı bir nokta üzerinde veri erişim süreçleri devam ettirilebilmektedir. Sağlanan olanaklar ile Disaster Recovery sürecine destek verilebilmektedir.
Backup	Windows Azure Blob Storage içerisinde depolanan veriler, çeşitli Windows Azure hizmetleri ile yedekle olanakları sağlanabilmektedir.
Geo-Replication	Depolanan verinin farklı coğrafik nokta da kopyaların oluşturulabilmektedir.
Erişim Protokolleri	Windows Azure Blob Storage , üzerinde taşıdığı verileri Http ya da Https olarak erişilebilmesinin olanak sağlayabilmektedir. Depolama çözümü 2616 RPC standart da bağlı olarak Http ya da Https üzerinde REST iletişimine de olanak sağlamaktadır.
Logging	Windows Azure Blob Storage üzerinde yapılan işlemlerin geçmiş dönük incelebilmesi amacı ile yapılan veri hareketleri depolanabilmektedir.
En Fazla Depolama Miktarı	Windows Azure Blob Storage, Block Blob olarak en fazla 200GB ve Page Blob olarak ise, 1TB veri depolayabilmektedir.

Geliştirilen iş uygulamaları, gerçekleştireceği iş sürecine bağlı olarak, depolama planlamaları yapılmaktadır. Yapılan depolama planları, iş verisinin okuma / yazma yoğunlukları ya da veri boyutları ile ilgili şekillenmektedir. **Windows Azure Blob Storage** konsepti ile iş gereksinimlerine göre **Page Blob** ve **Block Blob** isimli iki farklı nesne ile hizmet vermektedir. Aşağıda söz konusu nesneler ile ilgili temel bilgi karşılaştırmaları bulunmaktadır.

Özellik	Block Blob	Page Blob
En fazla Veri Depolama	4MB	512Byte
En Az Veri Depolama	200GB	1TB
Snapshot	Destekliyor	Destekliyor
Erişim Seviyesi	Private,Public blob, Public container	Private,Public blob, Public container
Erişim Protokolleri	Http/Https/Rest	Http/Https/Rest
Logging	Destekliyor	Destekliyor

Yapılan anlatımın anlaşılması amacı ile “**WindowsAzure.FunnyApp**” isimli uygulama örneği hazırlanmıştır. Hazırlanan uygulama örneği ile iş senaryoları incelenerek, **Windows Azure Platform** hakkında farklı deneyimler paylaşılmıştır.

Uygulama örneğinin de **Windows Azure Storage** ile ilişkili nesnelerin kullanımı kolaylaştırıcı anahtar değişkenler tanımlanmıştır. Tanımlanan değişkenler **Windows Azure Storage** içerisinde kullanılan nesnelerin yönetimini kolaylaştırması amaçlanmıştır.

```
public class Utils
{
    public const string ConfigurationString = "DataConnectionString";

    public const string CloudQueueKey = "imagequeue";
    public const string CloudBlobKey = "imageblob";
}
```

Uygulama Senaryosu - |

Kullanıcı tarafından yayınlanması istenen fotoğraf ile ilgili bilgileri girilerek, içeriğin uygulamaya yükleme süreci gerçekleşmektedir. Yükleme süreci kullanıcı tarafından fotoğrafın yüklenmesi ile **Windows Azure Table Storage** üzerine ilgili bilgilere yazılarak ve fotoğraf içeriğinin **Windows Azure Blob Storage** üzerine yüklenmesi ile tamamlanmaktadır. Gerçekleşen süreçler ile ilgili kaynak kodlar ve yorumları aşağıda bulunmaktadır.

```
public partial class ImageUploadPage : Page
{
    private static readonly object _look = new object();
    private static bool _storageInitialized = false;

    // işlem gerçekleştirecek olan nesnelerin tanımlanması
    private static CloudBlobClient _blobClient;
    private static CloudQueueClient _queueClient;

    protected void Page_Load(object sender, EventArgs e)
    {
        this.Page.Title = "Image Uploads";
        if (IsPostBack) return;
        InitializeStorage();
    }

    protected void ButtonSave_Click(object sender, EventArgs e)
    {
        if (FileUploadImage.HasFiles & Page.IsValid)
        {
            // Blob nesnesine yüklenecek dosya için benzersiz bir isim oluşturulması
            string uniqueBlobName = string.Format("{0}/funnyimage_{1}{2}", Utils.CloudBlobKey,
                Guid.NewGuid().ToString(),
                Path.GetExtension(FileUploadImage.FileName));
        }
    }
}
```

```

// Blob nesnesi ile ilgili anahtar ve nesne örneğinin alınması
CloudBlockBlob blob = _blobClient.GetBlockBlobReference(uniqueBobName);

// Blob nesnesi üzerine yüklenecek, Blob nesnesine dosyasının türünün a
tanması
blob.Properties.ContentType = FileUploadImage.PostedFile.ContentType;

// Blob nesnesine dosyasının yüklenmesi
blob.UploadFromStream(FileUploadImage.FileContent);

FunnyAppRepository<Post> postRepository = new FunnyAppRepository<Post>(
);
FunnyAppRepository<Tag> tagRepository = new FunnyAppRepository<Tag>();

MembershipUser user = Membership.GetUser(Page.User.Identity.Name);
if (user != null)
{
    Post post = new Post
    {
        PostContent = TextBoxDescription.Text,
        PostTitle = TextBoxTitle.Text,
        State = false,
        UserId = user.ProviderUserKey.ToString()
    };

    string[] tags = TextBoxTag.Text.Split(';');
    foreach (string tag in tags)
    {
        if (!string.IsNullOrEmpty(tag))
        {
            tagRepository.Create(new Tag()
            {
                PostRowKey = post.RowKey,
                PostPartitionKey = post.PartitionKey,
                TagName = tag,
            });
            tagRepository.SubmitChange();
        }
    }

    postRepository.Create(post);
    postRepository.SubmitChange();

// Kuyruk nesneleri
CloudQueue queue = _queueClient.GetQueueReference(Utils.CloudQueueK
ey);

// mesaj içeriğinin oluşturulması
// mesaj içerisinden birden fazla bilgi olması sebebi ile "," karak
teri

// ile bilgiler birbirinden ayrılmıştır.
CloudQueueMessage message =
    new CloudQueueMessage(string.Format("{0},{1},{2}", blob.Uri,
        post.PartitionKey, post.RowKey));
// Mesaj kuyruğa eklenmiştir.

```

```

        queue.AddMessage(message);

        LabelResult.Text = "Uploaded";
    }
    else
    {
        LabelResult.Text = "Failed";
    }
    }
}

private void InitializeStorage()
{
    if (_storageInitialized)
    {
        return;
    }

    lock (_lock)
    {
        if (_storageInitialized)
        {
            return;
        }

        try
        {
            // hesap bilgilerinin alınması
            CloudStorageAccount storageAccount =
                CloudStorageAccount.FromConfigurationSetting(Utils.ConfigurationKey);

            // image blob taşıyıcısının oluşturulması
            _blobClient = storageAccount.CreateCloudBlobClient();
            CloudBlobContainer container =
                _blobClient.GetContainerReference(Utils.CloudBlobKey);
            container.CreateIfNotExist();

            // Blob taşıyıcısına ile ilgili erişim ayarlarının tanımlanması
            var permissions = container.GetPermissions();
            permissions.PublicAccess = BlobContainerPublicAccessType.Container;
            container.SetPermissions(permissions);

            // create queue to communicate with worker role
            _queueClient = storageAccount.CreateCloudQueueClient();
            CloudQueue queue = _queueClient.GetQueueReference(Utils.CloudQueueKey);

            queue.CreateIfNotExist();
        }
        catch (WebException exception)
        {
            Trace.Write(exception.Message);
        }

        _storageInitialized = true;
    }
}

```

```
}  
}  
}
```

Uygulama Senaryosu - ||

Uygulama örneği olan "**WindowsAzure.FunnyApp**" çalışması, kullanıcılarından aldığı fotoğraf içerikleri yeninde boyutlandırarak, yayınlamaktadır. Yapılan boyutlandırma işlem yoğunluğun dengelemek ve kullanıcı işlem sürecinin gerçekleştirebilmesi amacı ile **Worker Role** tasarlanmıştır. Tasarlanan **Worker Role, Windows Azure Blob Storage** üzerinde boyutlandırılması amaçlanan dosya okunarak, boyutlandırma işlemlerine tabi tutulmaktadır. Yapılan işlemler sonucunda içerik URL adresi oluşmaktadır. Gerçekleşen süreçler ile ilgili kaynak kodlar ve yorumları aşağıda bulunmaktadır.

```
public class WorkerRole : RoleEntryPoint  
{  
    // işlem gerçekleştirecek olan nesnelerin tanımlanması  
    private CloudQueue _queue;  
    private CloudBlobContainer _container;  
  
    public override void Run()  
    {  
        // Worker nesnesi içerisinde sürecin sürekli olarak gerçekleştirilmesi amacı ile  
        // sonsuz döngüye alınması gerekmektedir.  
        while (true)  
        {  
            try  
            {  
                // İşlem sürecinde kuyruk içerisinde bulunan mesajın dinleme işlemi  
                CloudQueueMessage message = _queue.GetMessage();  
                if (message != null)  
                {  
                    string[] messageArray = message.AsString.Split(new char[] { ',' });  
  
                    string outputBlobUri = messageArray[0];  
                    string partitionKey = messageArray[1];  
                    string rowkey = messageArray[2];  
  
                    // Kuyruk içerisinde okunan mesajı, döngü içerisinde tekrar okuması önemek amacı ile  
                    // mesaj Peek edilmektedir.  
                    _queue.PeekMessage();  
  
                    // kuyruk nesnesi üzerinde gelen Blob nesne adresinin  
                    // istenilen formatlarda filitrelenmesi  
                    string inputBlobUri =  
                        Regex.Replace(outputBlobUri, "([^\.\.]+)(\.\.[^\.\.]+)?$", "$1-myimage$2");  
  
                    // İşlem yapılması istenen Blob nesnenin kontrol eldilmesi  
                    _container.CreateIfNotExist();  
                }  
            }  
        }  
    }  
}
```

```

        // okuma ve yazma süreçlerin gerçekleşen olan Blob adreslerine
        bağlantıların oluşturulması
        CloudBlob inputBlob = _container.GetBlobReference(outputBlobUri
    );
        CloudBlob outputBlob = _container.GetBlobReference(inputBlobUri
    );

        // Blob nesnesi içerisinde bulunan resim içeriği okunması
        using (BlobStream input = inputBlob.OpenRead())
        using (BlobStream output = outputBlob.OpenWrite())
        {
            ProcessImage(input, output);

            // Blob nesnesi üzerinde çalışılan içerik ile ilgili Blob ü
            zerine etki etmesi işlemi
            output.Commit();
            // İşlem sonucu olarak üretilen içerisinde output Blob
            // içerik türünün atanması
            outputBlob.Properties.ContentType = "image/jpeg";
            // İşlem sonucu olan içeriğin attanan özellikleri Blob nesn
            esi üzerine etki etmesi
            outputBlob.SetProperties();

            FunnyAppRepository<Post> postRepository = new FunnyAppRepos
            itory<Post>();
            Post post = postRepository.Find(partitionKey, rowkey);

            // Boyutlandırma işlemine uğrayan fotoğraf içeriğin
            // ilgili entity nesnesine yansıtılması
            post.PostImage = outputBlobUri;
            post.State = true;
            postRepository.Update(post);
            postRepository.SubmitChange();

            _queue.DeleteMessage(message);
        }
    }
    catch (StorageClientException e)
    {
        Trace.Write(e);
    }
}

public override bool OnStart()
{
    // Worker üzerinde başlatılması amaçlanan varsayılan bağlantı sayısı
    ServicePointManager.DefaultConnectionLimit = 12;
    // Uygulama içerisinde bağlantı bilgilerinin alınma işlemi
    CloudStorageAccount.SetConfigurationSettingPublisher((configName, configSet
    ter) =>
        configSetter(RoleEnvironment.GetConfigurationSettingValue(configName)))
    ;
}

```

```

// Uygulama içerisinde bağlantı bilgilerinin alınma işlemi
var storageAccount =
    CloudStorageAccount.FromConfigurationSetting(Utils.ConfigurationString)
;

// kuyruk nesnesi ile ilgili referansların oluşturulması
CloudQueueClient queueStorage = storageAccount.CreateCloudQueueClient();
_queue = queueStorage.GetQueueReference(Utils.CloudQueueKey);

// kuyruk nesnesinin bulunup-bulunmadığı kontrollünün yapılması
_queue.CreateIfNotExist();

// Blob nesnesi ile ilgili referansların oluşturulması
CloudBlobClient blobStorage = storageAccount.CreateCloudBlobClient();
_container = blobStorage.GetContainerReference(Utils.CloudBlobKey);

// Blob nesnesinin bulunup-bulunmadığının kontrollünün yapılması
_container.CreateIfNotExist();

return base.OnStart();
}

// Resim boyutlandırma işlemleri ile fonksiyon
public void ProcessImage(Stream input, Stream output) {. . .}
}

```

Windows Azure Blob Storage altyapısı, kullanımı ile ilgili iş süreçleri incelenmiştir. **Windows Azure Blob Storage** altyapısı **Page Blob** ve **Block Blob** depolama konsepti ile iş süreçlerine cevap üretmektedir. Depolama konseptleri, iş uygulamalarının, iş verisi üzerinde en performanslı şekilde çalışabilmesini olanak sağlamaktadır. Belirtilen konseptlerin tercihi ve uygulama şekilleri ile ilgili örnek senaryolar aşağıda bulunmaktadır.

Windows Azure Blob Storage, Page Blob

İş uygulamaları, içerisinde bulunduğu sürece bağlı olarak, yoğun veri okuma süreçlerin de bulunmaktadır. Örneğin; Yoğun video yayının yapan iş uygulamaları, istemcilerini besleyebilmek için talep edilen video içerikleri yoğun okuma işlemlerini gerçekleştirmesi gerekmektedir. Söz konusu süreç de yoğun okumalara en iyi performans yakalayabilmesi amacı ile **Windows Azure Page Blob** konseptinin tercih edilmesi daha sağlıklı sistem altyapısı oluşturmanıza olanak sağlayacaktır.

Windows Azure Blob Storage, Block Blob

İş gereksinimleri zaman içerisinde paralel olarak, yoğun dosya yükleme işlemleri gerektirebilmektedir. Örneğin; E-ticaret sistemleri ürün içeriklerini güncellemek amacı ile günlük içerik düzenleme süreçleri gerçekleştirmektedir. Gerçekleştirilen işlemler ile birçok ürün fotoğraf içerikleri yüklenmektedir. İzlenen sürecin en performanslı olarak

gerçekleřtirmek amacı ile **Windows Azure Block Blob** konseptinin tercih edilmesi başarılı sonuçlar alınmasını sağlayacaktır.

*Not: Yapılan anlatımlarda “**WindowsAzure.FunnyApp**” uygulama örneęi kullanılmıřtır. Uygulama ile ilgili kaynak kodları ařaęıdaki baęlantıyı kullanarak edinebilirsiniz.*

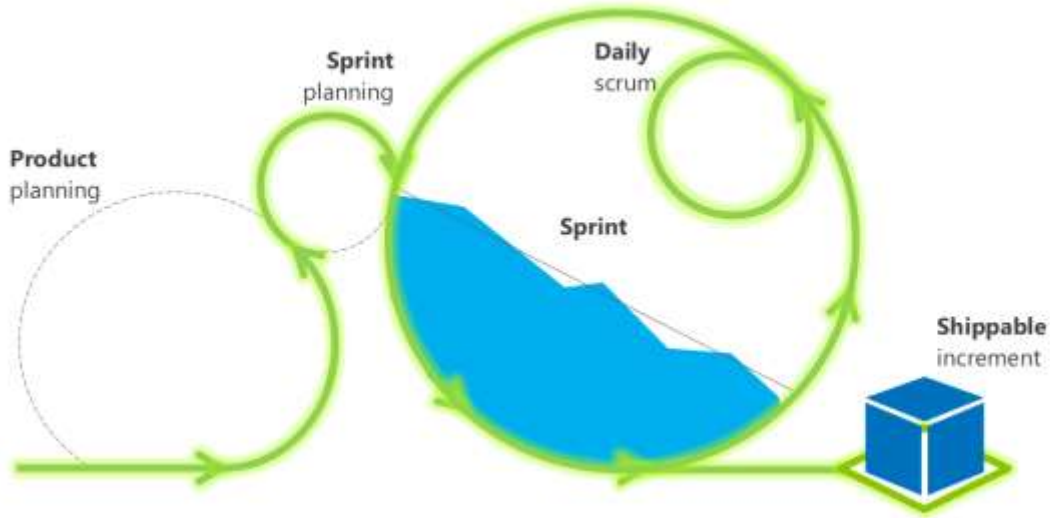
Github / <https://github.com/ibrahimatay/WindowsAzure.FunnyApp>

İř uygulamaların başarılı ve performanslı olarak alıřabilmesi için geliřtiricilerin, iř tecrübelerine göre eřitli yaklařımlar uygulanmaktadır. Günümüzde depolama ve verinin kullanımı konusunda yeni konseptler oluřturulmaktadır. Microsoft **Windows Azure Platform** ile de sürece yeni bir bakıř ası oluřturulmuřtur.

EK Not: Bölüm ile ilgili uygulama videosu ve sunumunu incelemek için ařaęıdaki baęlantıyı kullanabilirsiniz.

[http://www.ibrahimatay.org/post/2013/8/10/Windows-Azure-ile-Cloud-Computing-Uygulamalari---8-\(Video\).aspx](http://www.ibrahimatay.org/post/2013/8/10/Windows-Azure-ile-Cloud-Computing-Uygulamalari---8-(Video).aspx)

Cloud service lifecycle - Agile



Windows Azure ile Cloud Computing Uygulamaları – 9

İş coğrafyaları ve gereksinimleri her gün farklılaşıyor. Süreçlere uyum sağlamak her zamaninkinden zor ve maliyetli olmaktadır. Günümüz şartlarında şirketlerin ekonomik, kaliteli ve sürdürülebilir altyapılar sahip olmasının en kolay yolu Cloud Computing den geçmektedir. Bu bölümde Windows Azure Platform üzerine Windows Azure Cloud Services konsepti ile geliştirilmiş "WindowsAzure.FunnyApp" uygulaması adım adım yayınlama süreçlerini incelenmiştir.



Zaman hızlı ilerliyor. İş ihtiyaçları sürekli değişiyor. Bugün kullanılan iş uygulamaları yarın yeniden planlanması gerekebiliyor. İçerisinde bulunduğumuz dönem de ise iş uygulamalarının her zamankinden daha fazla esnek olması gerekmektedir.

Şirketler büyümeye ve farklı müşteri portföylerine ulaşmaya devam ediyor. Geçmişten sadece Türkiye pazarında çalışan şirketler, günümüzde Avrupa, Amerika ya da Afrika gibi bölgeler de çalışır hale geldi. Büyüyen şirketler, müşterileri ile sağlıklı iletişimi kurabilmesi için teknolojik altyapılarını, iş gereksinimlerine göre şekillendirmeleri gerekmektedir.



Sektörle olarak fark etmeksizin, iş coğrafyaları artıyor. Artan iş coğrafyaları, şirketlerin yeni müşterilerine ulaşabilmesi için yeni altyapı yatırımların yapmasını gerektirmektedir. Yapılan altyapı yatırımların başında ise, Web tabanlı uygulamaları gelmektedir.

Günümüzde şirketlerin müşterilerin her alanda ulaşma ve destek olma istediği, beraberinde altyapı sistemlerinin esnek, ekonomik ve sürdürülebilir ihtiyaçlarını doğurmuştur. Meydana gelen gereksinimler, teknoloji bilgi ve tecrübeler sonucunda **Cloud Computing** altyapısının oluşmasına neden olmuştur. **Cloud Computing** altyapısı çeşitli şirketler tarafında yorumlandığı gibi Microsoft tarafından da **Windows Azure Platform** ürünü ile kullanıcılarına birçok olanaklar sunmaktadır. Microsoft, **Windows Azure Platform** ürünü ile kolay yönetebilir, güvenilir **SLA(Service Level Agreement)** belgeli ve beraberinde farklı uygulama ve iş konseptleri sunmaktadır.

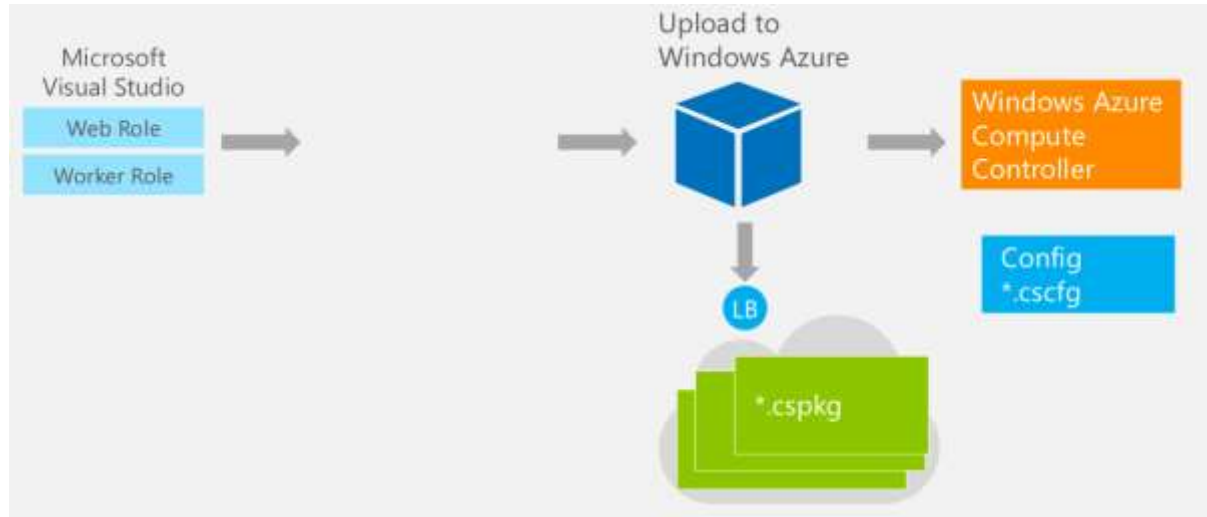
Günümüzde iş ihtiyaçlarına göre hızlı şekillenebilen şirketler değer kazanmaktadır.

Microsoft, **Windows Azure Platform** altyapısı ile şirketleri sahip oldukları uygulamaları esnek ve şuanda Avrupa, Amerika, Asya ya da gelecek de açılacak diğer veri merkezleri ile bölgeler den bağımsız olarak çalışabilmesine olanak sağlamaktadır.

Windows Azure Platform altyapısının incelenmesi ve **Windows Azure Cloud Services** konseptinin anlaşılması amacı ile "WindowsAzure.FunnyApp" uygulama çalışması geliştirilmiştir. Anlatımın devamında "WindowsAzure.FunnyApp" uygulamasının **Windows Azure Platform** içerisinde yayınlama süreci hakkında bilgiler verilecektir.

Windows Azure Cloud Service konsepti, esnek(genişleyebilen), güvenli ve **Agility** uygulama geliştirme süreçlerin de uygulanabilir uygulama konseptidir. **Windows Azure Cloud Services** konsepti ile hazırlanmış uygulamalar, **Windows Azure Platform** 'na paketlenerek, yayınlanmaktadır.

Cloud Computing uygulamalarının esnek çalışabilme yani genişleyebile süreçleri ile ilgili "[Windows Azure ile Cloud Computing Uygulamaları - 2](#)" makalesini incelemenizi tavsiye ederim.



Windows Azure Platform içerisinde yayınlanması istenen **Windows Azure Cloud Service** uygulamaları, platform içerisinde **Production** ve **Staging** olarak iki ayrı deployment tekniği kullanarak yayınlanabilmektedir. Söz konusu deployment teknikleri ile ilgili aşağıda bilgi verilmiştir.

Staging

İş uygulamaları geliştirme yaklaşımına bağlı olarak, manuel test yapılması gerektirdiği durumlar ile karşılaşılabilmektedir. Söz konusu durumda uygulamalar test ortamında deploy edilerek, amaçlanan testler de istenilen sonuçların alınması ile ürün ortamına taşınmaktadır.

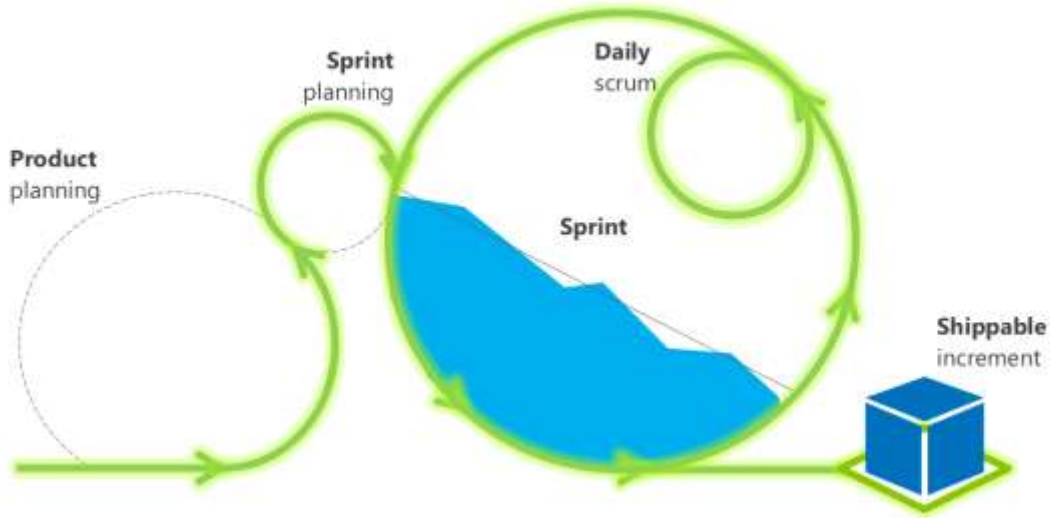
Windows Azure Cloud Service deployment sürecinde uygulama ilk olarak **Staging** olarak deploy ile manuel testlerin yapılması ve uygulama kararlılığının anlaşılması ile sadece "**Swap**" butonu kullanarak, ürün ortamına taşınabilmektedir.

Production

İş uygulamaları içerisinde bulunduğu durumlara göre direk son kullanıcın karşısına çıkarılması gerekmektedir. Yapıla çalışma ile kullanıcı direk olarak, son kullanıcın karşısına çıkarılmaktadır.

Windows Azure Cloud Service konsepti ile geliştirilen iş uygulamalarını direk olarak, son kullanıcın karşısına çıkarabilecek şekilde yayınlanması istendiğın de uygulama **Production** deployment tekniğı kullanarak yayınlanabilmektedir.

Cloud service lifecycle - Agile



Agility uygulama geliştirme yaklaşımları kullanarak geliştirilen yazılım projeleri, **Spint** denilen zaman periyotları kullanarak geliştirmektedir. Geliştirme sürecinde her **Spint** sonunda UAT süreci gerçekleştirilerek, uygulama kararlılığı test edilmektedir. Yapılan test sonuçlarına bağlı olarak uygulama ürün ortamına taşınmaktadır. **Windows Azure Cloud Service** konsepti ile geliştirilen uygulamalar, **Spint** sonunda "**Staging**" deployment yapılarak, kullanıcı kabul testleri yapılabilir. Yapılan testler sonucun da karalı sürüm elde edilmesi ile "**Swap**" butonu kullanarak, uygulama kolayca ürün ortamına taşınabilmektedir.

Windows Azure Cloud Services konsepti ile hazırlanan uygulamalar, çeşitli yayınlama türleri kullanarak yayınlanabilmektedir. Söz konusu yayınlama türleri aşağıda belirtilmiştir.

- **Windows Azure Platform** Yönetim Portal Deployment alanlarını kullanarak
- Kaynak kontrol sistemleri kullanarak(TFS, TFS Services, Git, ve Local Git)
- Powershell Script kullanarak
- Visual Studio 20[08,10,12,13] kullanarak

Yapılan anlatım sürecinin devamında "[Windows Azure ile Cloud Computing Uygulamaları](#)" yayın serisi için hazırlanan "[WindowsAzure.FunnyApp](#)" uygulaması, Visual

Studio 2012 kullanarak Deployment ve yayınlama süreci inceleniyor olacaktır. Gerçekleştirilecek adımlar aşağıda bulunmaktadır.

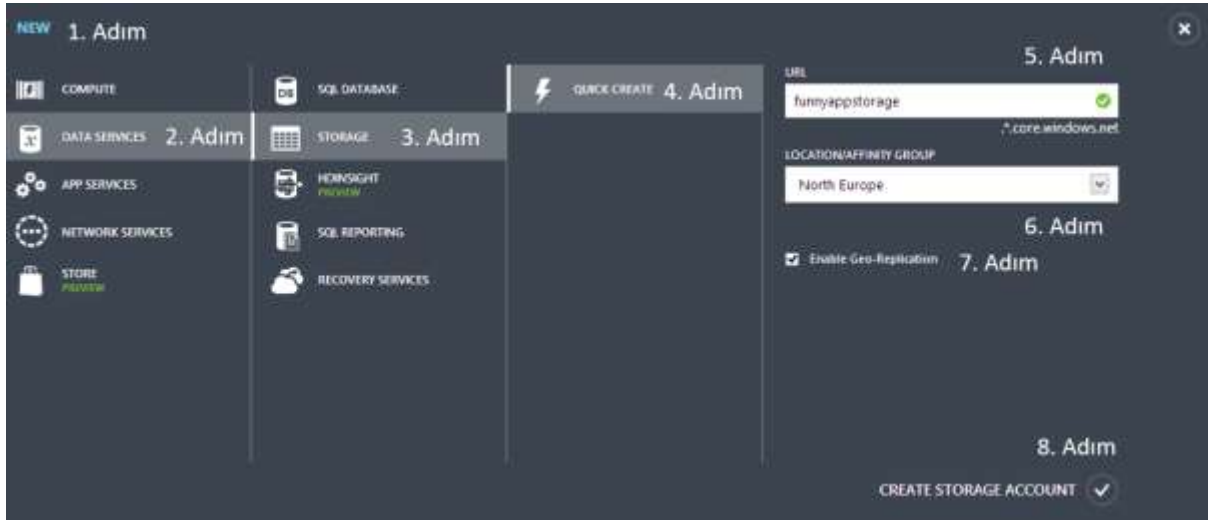
1. Uygulama ile ilgili **Windows Azure Storage** hesabının(örneği "FunnAppStorage") oluşturulması
2. Uygulama ile ilgili **Windows Azure Cloud Service** hesabının(Örneğin "FunnyApp") oluşturulması
3. Uygulama konfigürasyonunun sağlanması
4. Visual Studio 2012 yardımı ile uygulama **Windows Azure** adımların başlatılması
5. Deployment yapılan uygulamanın "**Swap**" butonu kullanarak, ürün ortamına alınması

Uygulama deployment ve yayınlama süreci ile ilgili anlatımlar da kullanılacak olan "WindowsAzure.FunnyApp" uygulaması ile ilgili olarak ayrıntılı bilgi için "[Windows Azure ile Cloud Computing Uygulamaları - 5](#)" makalesini incelemenizi tavsiye ederim.

1 - Uygulama ile ilgili Windows Azure Storage hesabının oluşturulması

Windows Azure Storage, Windows Azure Platform içerisinde uygulamaların depolama gereksinimleri sağlanabilmesi amacı ile oluşturulan modern depolama konseptidir.

Windows Azure Platform içerisinde **Windows Azure Storage** hesabı oluşturulabilmek için aşağıda bulunan resimdeki adımlar gerçekleştirebilirsiniz.

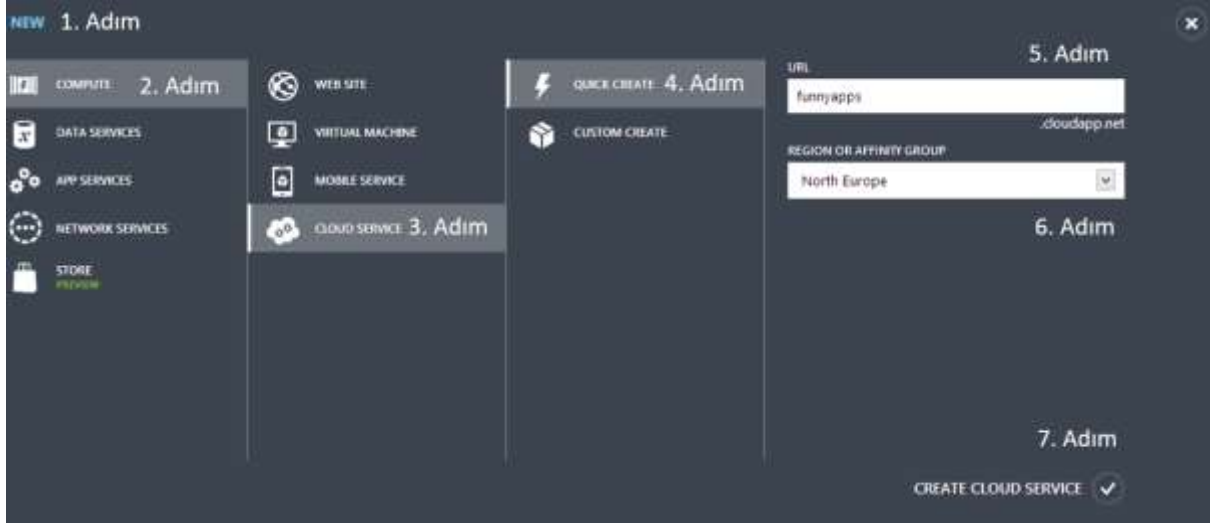


Windows Azure Storage konsepti ile ilgili detaylı teknik bilgi için "[Windows Azure ile Cloud Computing Uygulamalar - 7](#)" ve "[Windows Azure ile Cloud Computing Uygulamalar - 8](#)" makalelerinin incelemenizi tavsiye ederim.

2 - Uygulama ile ilgili Windows Azure Cloud Service hesabının oluşturulması

Windows Azure Cloud Services, Windows Azure Platform içerisinde uygulamaların esnek, performanslı ve yüksek güvenlik düzeylerinde çalışabilmesine olanak sağlayan modern uygulama yayınlama konseptidir. **Windows Azure Platform** içerisinde **Windows Azure**

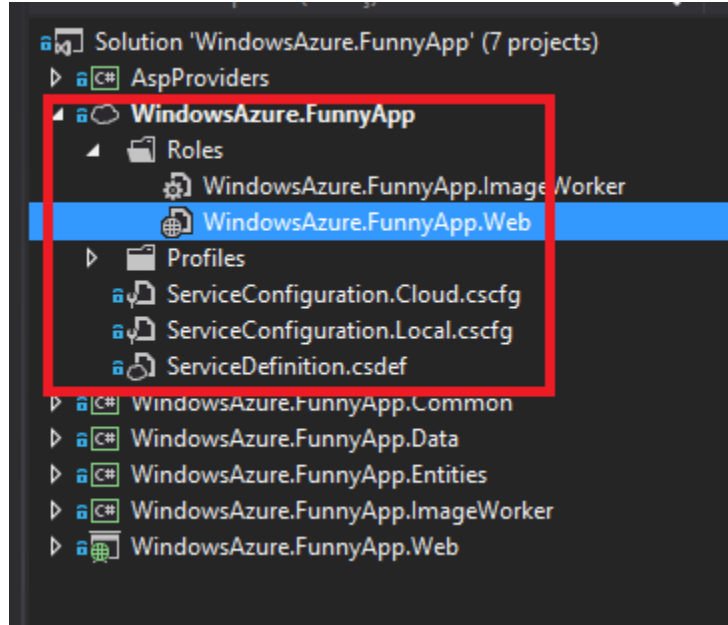
Cloud Service hesabı oluşturulabilmek için aşağıda bulunan resimdeki adımları izleyebilirsiniz.



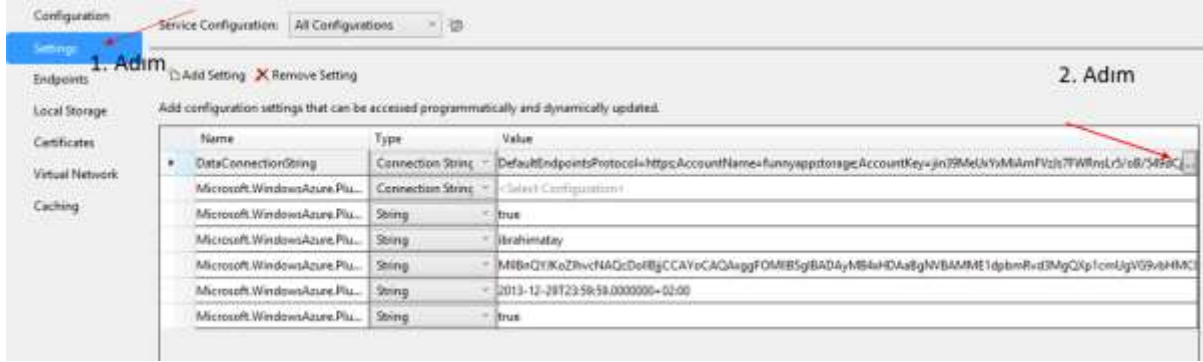
Windows Azure Cloud Services konsepti ile ilgili teknik bilgi için "[Windows Azure ile Cloud Computing Uygulamalar - 5](#)" ve "[Windows Azure ile Cloud Computing Uygulamalar - 6](#)" makalelerinin incelemenizi tavsiye ederim.

3 - Uygulama konfigürasyonunun sağlanması

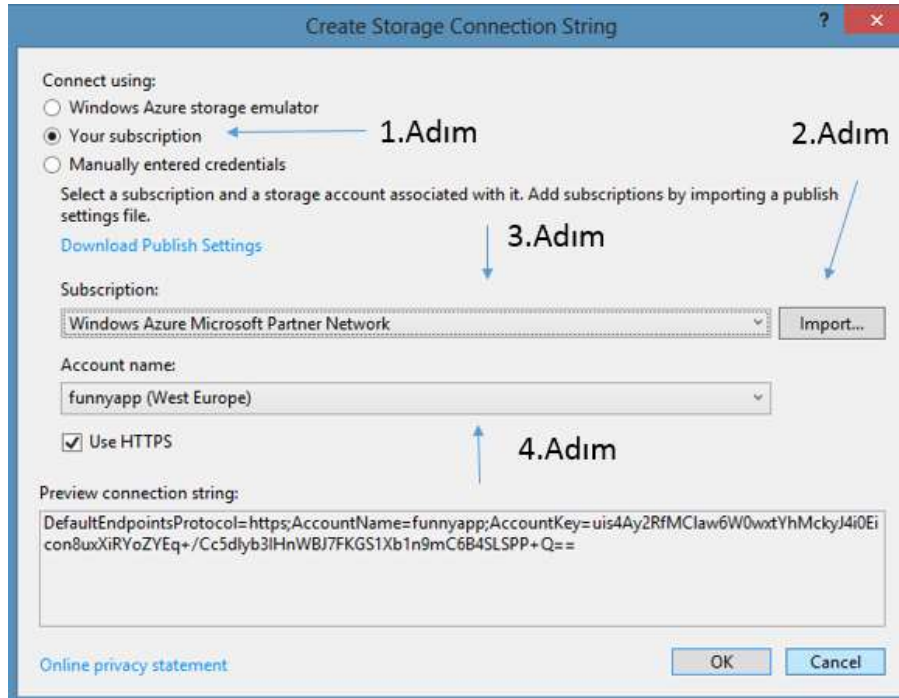
Windows Azure Platform içerisinde **Windows Azure Cloud Service** olarak geliştirilen uygulamanın bağımlı olduğu **Windows Azure Platform** hizmetleri ile ilgili iletişimini sağlayabilmesi amacı bazı bilgilere ihtiyaç duymaktadır. Uygulama örneği olan, "WindowsAzure.FunnyApp" 'ın **Windows Azure Storage** ile ilgili hesap bilgileri aktarılabilmesi amacı ile aşağıdaki adımları izleyebilirsiniz.



Uygulama çalışması olan “WindowsAzure.FunnApp” projesi içerisinde bulunan “WindowsAzure.FunnyApp” proje şablonu içerisinde bulunan “Roles” bölüne gelinmesi gerekmektedir. “Roles” bölümünde bulunan “WindowsAzure.FunnyApp.Web” role seçilerek, konfigürasyon alanı açılmaktadır. **Windows Azure Cloud Service**, Role konfigürasyon alanına ulaşabilmek için aşağıdaki adımları izlenmesi gerekmektedir.



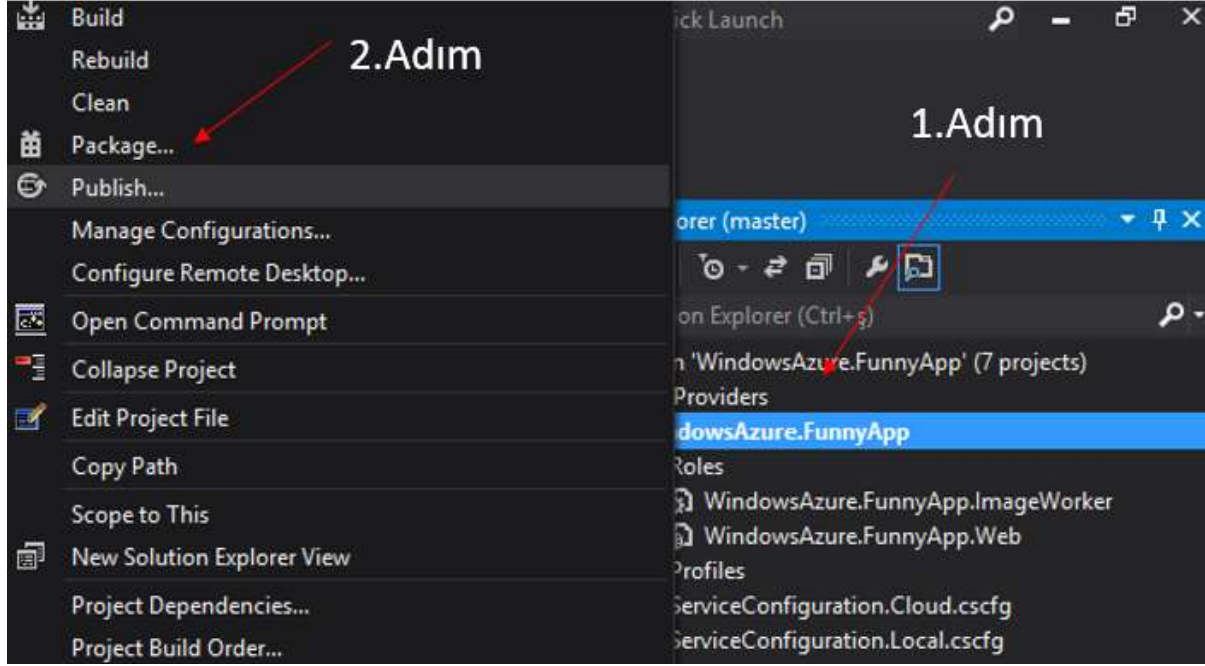
Uygulama çalışmasının kullanmış olduğu **Windows Azure** hizmetleri ile bağlantısının sağlanabilmesi amacı ile sahip olduğunuz **Windows Azure** hesap bilgisine ihtiyaç duymaktadır. Hesap bilgisinin uygulama içerisine aktarılabilmesi için resimdeki adımları izleyebilirsiniz.



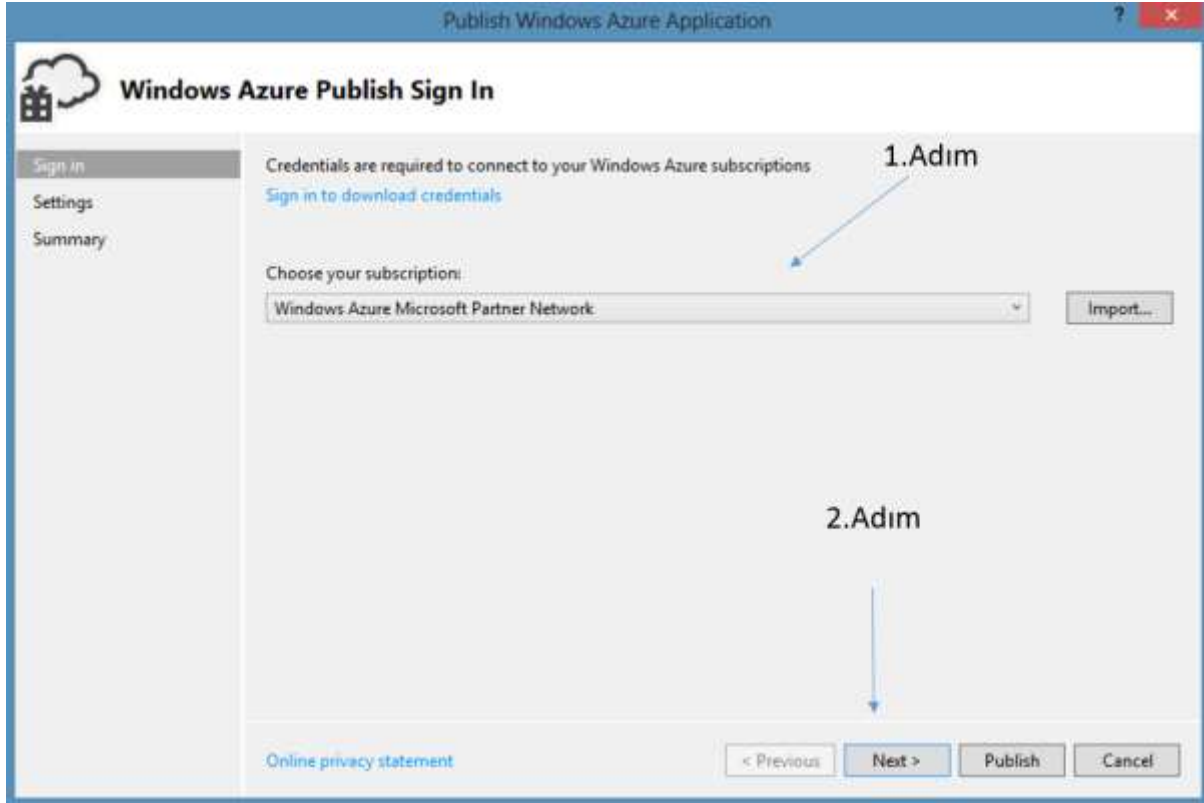
Yukarıda belirtilen adımlar ile sahip olunan **Windows Azure** hesap bilgilerinin, seçilmesi ve devamında içeriye aktarılması gerekmektedir. Gerçekleştirilen işlem adımlarında sahip olunan **Windows Azure** hesabı ile ilişkili “Publish Setting” dosyası kullanılmaktadır. Eğer böyle bir dosyanız yok ise, “Download Publish Settings” bağlantısını kullanarak edinebilirsiniz.

4 - Visual Studio yardımı ile uygulama Windows Azure adımların başlatılması

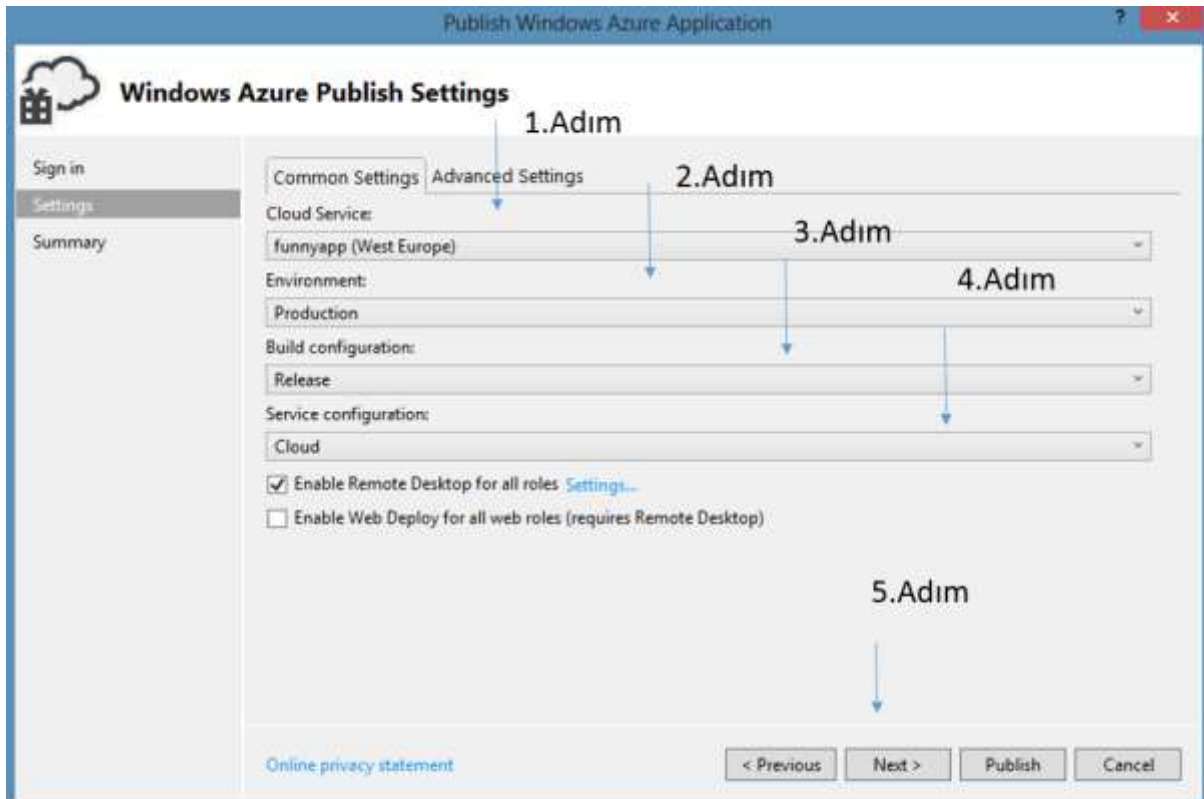
Hazırlanan “WindowsAzure.FunnyApp” uygulaması, **Windows Azure Cloud Service** uygulaması olarak yayınlanma süreci ile ilgili aşağıdaki adımları izleyebilirsiniz.



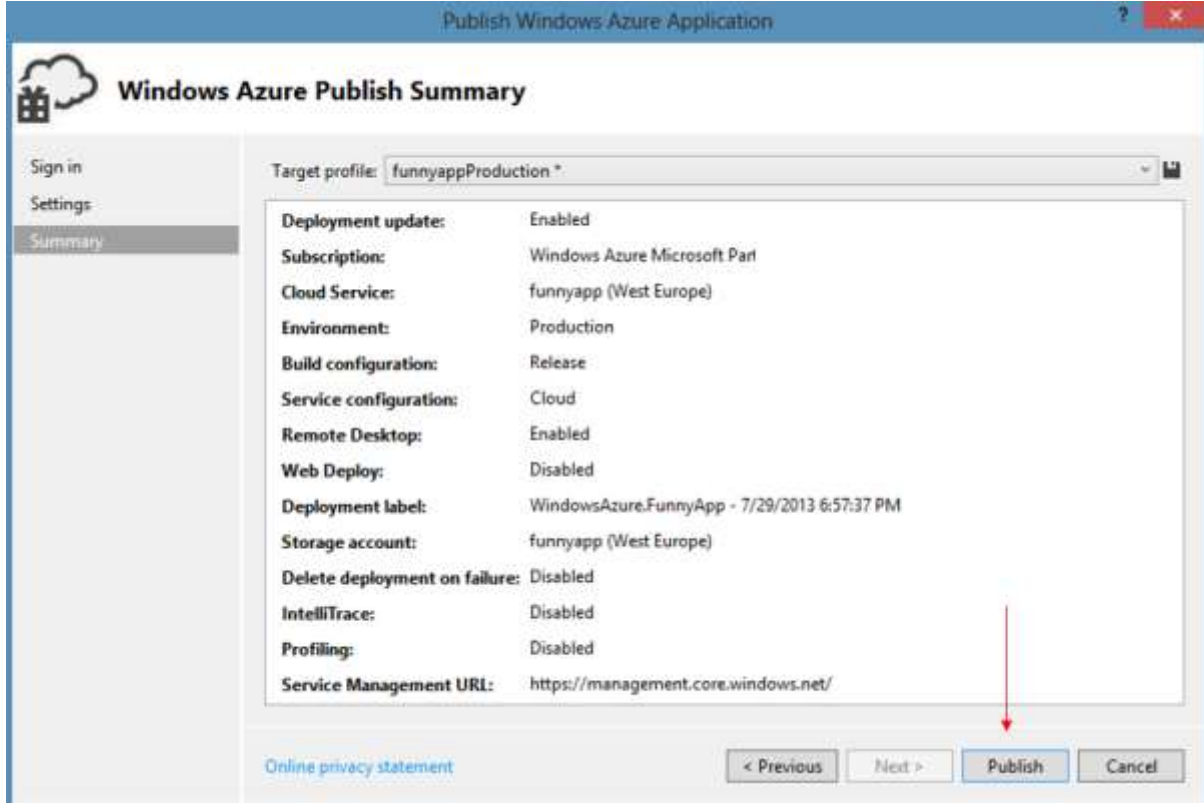
Uygulama yayınlama süreci ile ilişki **Windows Azure** hesabı ilişkilendirme işlemleri ile ilgili aşağıdaki adımları izleyebilirsiniz.



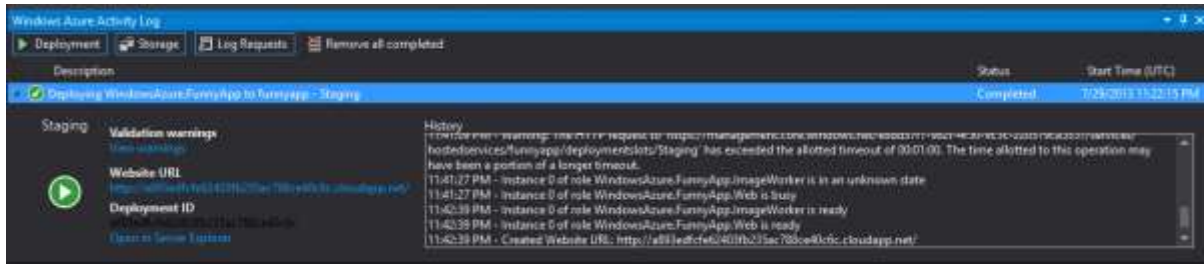
Uygulama yayınlaması istenen **Windows Azure** hesabı ile ilgili uygulama ortamlarının ilişkilendirilmesi ile ilgili aşağıdaki adımları izleyebilirsiniz.



Windows Azure hesap ilişkilendirilmesi ve uygulama konfigürasyonun yapılması ile uygulama **Windows Azure Platform** içerisinde yayınlanabilir hale ulaşmıştır. Aşağıdaki adımları izleyerek, uygulama yayınlama işlemleri başlatılabilmektedir.



Uygulamanın **Windows Azure Platform** yayınlama işlemi ile ilgili gerçekleşen süreci "**Windows Azure Tools for Microsoft Visual Studio**" ile birlikte gelen, "**Windows Azure Activity Log**" panel kullanarak izleyebilirsiniz.



Uygulama "**Windows Azure Tools for Microsoft Visual Studio**" kullanarak, "**Staging**" olarak yayınlanmıştır. Yayınlanan uygulamanın, son kullanıcı ortamında farklı ve benzersiz bir site adresi üzerinde yayın yapmaktadır.

5 - Deployment "Staging" yapılan uygulama, "Swap" butonu kullanarak, Production ortamına alınması

Windows Azure Platform içerisinde "**Staging**" olarak yayınlanan "**WindowsAzure.FunnyApp**" uygulaması, amaçlanan testlerin gerçekleştirilmesi ile

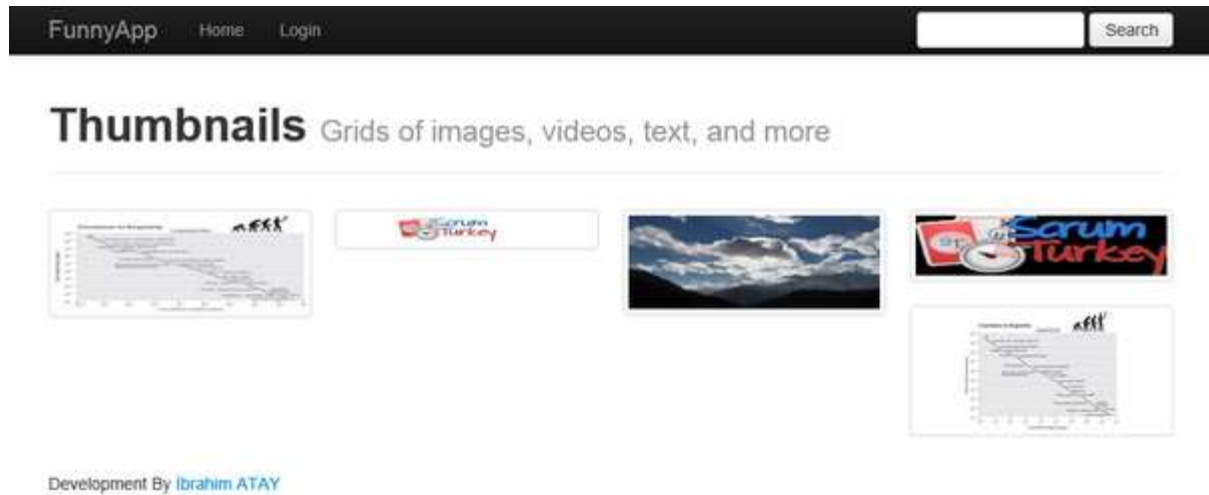
“Swap” butonu kullanarak, ürün ortamına taşınabilmektedir.

funnyapp

DASHBOARD	MONITOR	CONFIGURE	SCALE	PREVIEW	INSTANCES	LINKED RESOURCES	CERTIFICATES
PRODUCTION		STAGING					
NAME	STATUS		ROLE		SI		
WindowsAzure.FunnyApp.ImageWorke...	✓	Running	WindowsAzure.FunnyApp.Image...		S		
WindowsAzure.FunnyApp.Web_IN_0	✓	Running	WindowsAzure.FunnyApp.Web		Sr		



Belirtilen işlemlerin yapılması sonucunda uygulama son kullanıcının karşısına çıkabilecek, kullanıcı dostu adresi olan bir **Windows Azure Cloud Service** uygulamayı yayınlanmış bulunmaktadır.



Yapılan işlemler sonucunda makale serisi uygulama örneği uygulaması olan “WindowsAzure.FunnyApp”, **Windows Azure Platform** üzerinde **Windows Azure Cloud Services** konsepti ile yayınlanmış bulunmaktadır.

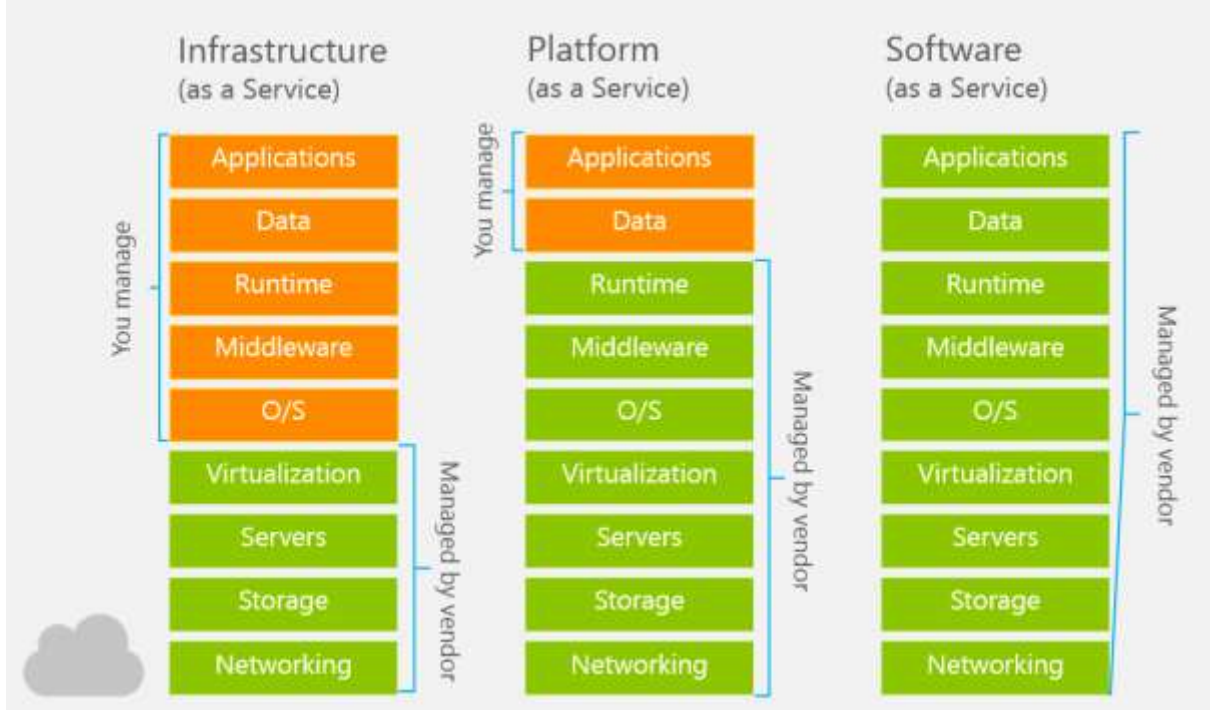
Not: Yapılan anlatımın örneklenmesi amacı ile “WindowsAzure.FunnyApp” uygulaması hazırlanmıştır. Aşağıdaki bağlantı kullanarak, uygulama kaynak kodlarına erişebilirsiniz.

Github / <https://github.com/ibrahimatay/WindowsAzure.FunnyApp>

İş coğrafyaları ve gereksinimleri her gün farklılaşıyor. Süreçlere uyum sağlamak her zamanınkinden zor ve maliyetli olmaktadır. Günümüz şartlarında şirketlerin ekonomik, kaliteli ve sürdürülebilir altyapılar sahip olmasının en kolay yolu **Cloud Computing** den geçmektedir. Bu yazılı ile **Cloud Computing** altyapısı olan **Windows Azure Platform** üzerine **Windows Azure Cloud Services** konsepti ile geliştirilmiş “WindowsAzure.FunnyApp” uygulaması adım adım yayınlama süreçlerini incelemiş olduk.

EK Not: Bölüm ile ilgili uygulama videosu ve sunumunu incelemek için aşağıdaki bağlantıyı kullanabilirsiniz.

[http://www.ibrahimatay.org/post/2013/8/12/Windows-Azure-ile-Cloud-Computing-Uygulamalari---9-\(Video\).aspx](http://www.ibrahimatay.org/post/2013/8/12/Windows-Azure-ile-Cloud-Computing-Uygulamalari---9-(Video).aspx)



Windows Azure ile Cloud Computing Uygulamaları – 10

İş coğrafyaları ve gereksinimleri her gün farklılaşıyor. Süreçlere uyum sağlamak her zamanınkinden zor ve maliyetli olmaktadır. Günümüz şartlarında şirketlerin ekonomik, kaliteli ve sürdürülebilir altyapılar sahip olmasının en kolay yolu Cloud Computing den geçmektedir. Bu bölümde Windows Azure Platform üzerine Windows Azure Cloud Services konsepti ile geliştirilmiş "WindowsAzure.FunnyApp" uygulaması adım adım yayınlama süreçlerini incelenmiştir.



Zaman hızlı ilerliyor. Hedefe giden yollar farklılaşıyor. Her gün yeni proje ve girişimler ile ilerlemeye devam ediyoruz. Yaşam kültürleri değişiyor. Değişen kültürler, beraberinde yeni gereksinimler ve sektörler oluşturmaktadır. Üretici den tüketiciye uzanan yolda ne hızı ulaşım ise teknoloji den geçmektedir.



Günümüz iş dengesin de internet odaklı hizmet ve ürünlerin artması, ürün ya da hizmetin sunum sürecinde kullanılan altyapılarının önemini arttırmaktadır. Altyapıların kesintisiz, en az maliyet ve müşteri yoğunluklarına cevap üretmesi gereksinimleri bulunmaktadır.

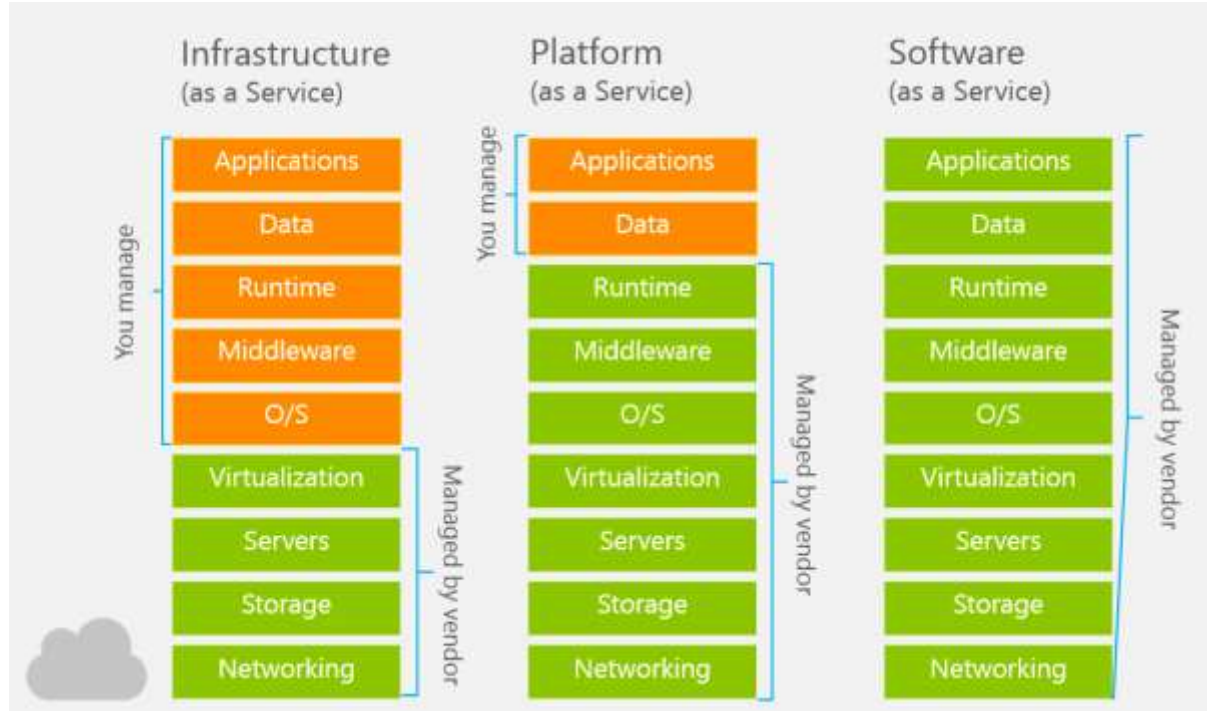
Serbest ticaret pazarlarındaki ürün satışının büyük bölümü internet üzerinde gerçekleştirilmektedir. Büyüyen internet tabanlı satış-pazarlama altyapıları, ıslak imza ile gerçekleştirilen süreçlerden daha güvenli olması gerekmektedir. Diğer bir önemli nokta ise, altyapıların farklı müşteri yoğunluklarında başarılı, cevap üretecek yeteneğinin olabilmesidir. Her şirketin, iş gereksinimleri gerçekleştirmek amacı ile şirketlere özel veri merkezlerin kurması, teknoloji dışı üretim yapan şirketlerin, yeni maliyet ve teknik uzmanların bulmasına problemlerine neden olmaktadır.

Teknoloji, müşteri gereksinimleri oluşmadan çözüm üretmeye devam etmektedir.

Değişmeye tek şeyin değişim olduğu düşünülen dünyamız da iş gereksinimlerin ekonomi, yoğunluk ya da başarı oranlarına göre değişmektedir. Yapılan her hata geleceğe için yeni ışık olmaktadır. Her yeni ışık ise, yeni adımlar ya da yeni teknoloji altyapı sistemleri olmaktadır.

Teknoloji ışığının en parlak olduğu dönemiz de iş çözüm üretmek, geçmişten daha kolay olmaktadır. İş çözümlerinin yaşam süreçleri incelendiğinde de çözümün yaşam sürecinde karşılaşılan problem ve gereksinimlerin üretim sürecinden daha fazla olduğu görülmektedir. İş çözümlerinde altyapı beklentilerinin sağlanabilmesi amacı ile **Cloud Computing** altyapısı oluşturulmuştur.

Cloud Computing altyapısı, iş çözümlerin beklenmedik yoğunluklara cevap üretebilecek, ekonomik ve iş devamlılığını sağlaması amacı ile hazırlanmıştır. Belirtilen altyapı özelliklerine ilgili olarak ise, birçok şirket tarafından incelenerek yorumlanmıştır. Özellikle Microsoft, **Windows Azure Platform** ürünü ile yüksek kullanıcı deneyimin bulunduğunu, kolay yönetilebilir olması ile öne çıkmaktadır.



Cloud Computing, kullanıcıların gereksinim duyacağı altyapıları hizmet olarak sunmaktadır. Sunulan her hizmet iş gereksinimlerin gereğince şekillenerek iş çözümü haline gelmektedir. **Cloud Computing Infrastructure as a Services (IaaS)**, **Platform as a Service (PaaS)**, **Software as a Service (SaaS)** ve yukarıdaki resimde belirtilmeyen birçok hizmet yaklaşımı ile kullanıcılarına iş çözümleri sunmaktadır.

Windows Azure Platform ve **Platform as a Service (PaaS)** uygulama modelinin anlaşılması amacı ile "[Windows Azure ile Cloud Computing Uygulamaları](#)" makale serisini hazırlanmıştır. Makale serisinin daha anlamlı olabilmesi için ise "WindowsAzure.FunnyApp" isimli uygulama örneği hazırlanmıştır. Uygulama örneği ile **Windows Azure Cloud Services**, **Windows Azure Storage** ve diğer **Windows Azure** hizmetleri hakkında çalışılmıştır.

Makale serisinde yapılan anlatımların örneklenmesi amacı ile hazırlanan "WindowsAzure.FunnyApp" uygulaması ile ilgili olarak, aşağıdaki bağlantıyı kullanarak kaynak kodları edinebilirsiniz.

WindowsAzure.FunnyApp / <https://github.com/ibrahimatay/WindowsAzure.FunnyApp>

Windows Azure Platform yeni güncellemeler ve eklenen ürünler ile gelişmeye devam etmektedir. Makale serisi ile uygulama geliştiricilerin **Windows Azure Platform** ile ilgili fikri edinerek, kolayca uygulama geliştirebilmeleri amaçlanmıştır.

EK Not: Bölüm ile ilgili uygulama videosu ve sunumunu incelemek için aşağıdaki bağlantıyı kullanabilirsiniz.

[http://www.ibrahimatay.org/post/2013/8/13/Windows-Azure-ile-Cloud-Computing-Uygulamalari---10-\(Video\).aspx](http://www.ibrahimatay.org/post/2013/8/13/Windows-Azure-ile-Cloud-Computing-Uygulamalari---10-(Video).aspx)

Kaynakça

Architectural Styles and the Design of Network-based Software Architectures

<http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>

Cloud Computing Wiki

http://en.wikipedia.org/wiki/Cloud_computing

MEET Windows Azure Konferansı Notları

<http://www.ibrahimatay.org/post/2012/6/14/MEET-Windows-Azure-Konferansi-Notlari.aspx>

Cloud Computing Düşüncesi

<http://www.ibrahimatay.org/post/2012/7/4/Cloud-Computing-Dusuncesi-Cloud-Computing-thought.aspx>

Scalability

<http://en.wikipedia.org/wiki/Scalability>

Scale-Out Vs Scale-Up

<http://highscalability.com/blog/2010/9/1/scale-out-vs-scale-up.html>

Windows Azure Training Kit November 2012

<http://www.microsoft.com/en-us/download/details.aspx?id=8396>

ISO 27001

<http://www.cert.sd/images/stories/iso27001.pdf>

Eticaret Kitabı - Şule ÖZMEN

<http://www.eticaretkitabi.com/>

Windows Azure Security

<http://www.windowsazure.com/en-us/support/trust-center/security/>

Guidance for Resilient Cloud Architectures

<http://msdn.microsoft.com/en-us/library/windowsazure/jj853352.aspx>

The NIST Definition of Cloud Computing

<http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf>

Unleashing the Potential of Cloud Computing in Europe

http://ec.europa.eu/information_society/activities/cloudcomputing/docs/com/com_cloud.pdf

Windows Azure Queues and Windows Azure Service Bus Queues - Compared and Contrasted

[http://msdn.microsoft.com/en-us/library/hh767287\(VS.103\).aspx](http://msdn.microsoft.com/en-us/library/hh767287(VS.103).aspx)

What is a cloud service?

<http://www.windowsazure.com/en-us/manage/services/cloud-services/what-is-a-cloud-service/>

The NIST Definition of Cloud Computing

<http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf>

TC Kimlik Numarası

http://www.nvi.gov.tr/Hakkimizda/Projeler,Mernis_Hedef.html

http://tr.wikipedia.org/wiki/T%C3%BCrkiye_Cumhuriyeti_Kimlik_Numaras%C4%B1

no:sql(east)

<https://nosqleast.com/2009/>

Repository Pattern

<http://martinfowler.com/eaCatalog/repository.html>

Windows Azure Table Storage and SQL Database - Compared and Contrasted

<http://msdn.microsoft.com/en-us/library/windowsazure/jj553018.aspx>

Windows Azure Table Storage

<http://www.windowsazure.com/en-us/develop/net/how-to-guides/table-services/>

Understanding Block Blobs and Page Blobs

<http://msdn.microsoft.com/en-us/library/ee691964.aspx>

RPC 2616

<http://www.ietf.org/rfc/rfc2616.txt>