9. Bölüm

Gereksinimler Yönetmek

Tanıtım

Gereksinim yönetimi, başlangıçtan teslimata kadar sistem gereksinimlerinin tanımlanmasını, belgelenmesini ve izlenmesini içerir. Bu tanımın özü, gereksinimlerin gerçek anlamını anlamak ve sistemin yaşam döngüsü boyunca müşteri (ve paydaş) beklentilerinin yönetimidir. Sağlam bir gereksinim yönetimi süreci, başarılı bir projenin anahtarıdır.

Hull et al. (2011), gereksinim yönetiminin beş ana zorluğu olduğunu öne sürmektedir. Birincisi, çok az sayıda kuruluşun iyi tanımlanmış bir gereksinim yönetimi sürecine sahip olması ve dolayısıyla bu kuruluşlardaki çok az kişinin gereksinim yönetimi deneyimine sahip olmasıdır. İkinci zorluk, kullanıcı veya paydaş gereksinimleri ve sistemleri arasında ayrım yapma zorluğudur. Üçüncü sorun, kuruluşların gereksinimleri farklı şekilde yönetmesi, deneyimin ve en iyi uygulamaların yayılmasını ve aktarılabilirliğini zorlaştırmasıdır. İlerlemenin izlenmesindeki zorluklar da başka bir sorundur. Ve son olarak, değişen gereksinimleri yönetmenin önemli bir zorluk olduğunu öne sürüyorlar. İyi tanımlanmış bir gereksinim yönetimi süreci oluşturmanın bu sorunları ele almanın anahtarı ol

Reinhart ve Meis (2012) üretim mühendisliğinden uyarlanan belirli gereksinim yönetimi yaklaşımlarının "eşzamanlı mühendisliğin", yani genellikle sırayla yürütülen eşit olmayan faaliyetlerin paralelleştirilmesinin yürütülmesini geliştirmek için nasıl kullanılabileceğini tartışır. Başarı faktörlerinin simülasyon prototipleme, mevcut bilginin entegrasyonu, sık iletişim ve araçların verimli kullanımını içerdiğini öne sürüyorlar.

Çoğu kuruluşta açık bir gereksinim yönetimi süreci yoktur, ancak bu, kuruluş içinde gereksinim yönetiminin gerçekleşmediği anlamına gelmez. Gereksinim uygulamaları muhtemelen organizasyonda örtük olarak mevcuttur, ancak bu uygulamalar genellikle belgelenmez. Herhangi bir kuruluşta gereksinim yönetimi sürecini iyileştirmenin ilk adımlarından biri, mevcut uygulamaları belgelemektir.

Konfigürasyon Yönetimi ve Kontrolü

Konfigürasyon yönetimi, sistemin önemli yapılarının tanımlanmasını, izlenmesini ve kontrolünü içerir. Gereksinim mühendisiyle ilgili yapılandırma öğeleri, bireysel gereksinimleri, gereksinim kaynaklarını, paydaşları ve gereksinim belirtim belgesini içerir. Öğeler yapılandırma altındayken, bu öğelerde değişiklikler yalnızca değişiklik yapmaya yetkili kişiler tarafından yapılabilir ve tüm değişiklikler izlenir, zaman damgalı ve sürüm damgalıdır. IEEE 29148, IEEE 12207 ve ISO 9000 gibi standartların tümü, sağlam bir yapılandırma yönetimi programının yürürlükte olmasını ve tüm gereksinim yapılarının yapılandırma kontrolü altına alınmasını gerektirir.

Yapılandırma kontrolü, gereksinim değişikliklerinin talep edilmesini, değerlendirilmesini ve onaylanmasını, yayınların onaylanmasını içerir. Tüm gereksinim yapılarını yapılandırma kontrolü altına almak, eski ve saçma gereksinimler yaratma olasılığını azaltmaya yardımcı olur.

Bu nedenle gereksinim yönetiminin önemli bir parçası, disiplinli konfigürasyon yönetimi kontrol süreçleridir. Başarılı bir konfigürasyon yönetimi için gerekli disiplinin uygulanmasında, uygun işlevsellik içeren gereksinim yönetimi, konfigürasyon yönetimi veya proje yönetimi araçları esastır.

Farklılıkları Uzlaştırma

Gereksinim yönetimindeki en önemli faaliyetlerden biri, özellikle aşağıdakileri oluştururken veya keşfederken fikir birliği oluşturmaktır:

Misyon ifadeleri
Hedefler
Gereksinimler
Sıralamalar

Ancak paydaş grupları arasında fikir birliğine varmak kolay değildir. Tuckman'ın ekip oluşturma teorisi, grup oluşumunun bir "oluşturma, fırtına, norm oluşturma, gerçekleştirme ve erteleme" modelini izlediğini öne sürer. öncül şu ki

bir ekip, uyumlu olmayan bir grubu yüksek işlevli bir gruba önemli ölçüde değiştirebilir. İlgilenen okuyucular, Sidney Lumet'in 1957 tarihli filmi 12 Kızgın Adam'ı izlemek isteyebilir . Aslında jürinin Tuckman takım oluşum dizisinden geçtiğini görebilirsiniz ve bu aynı zamanda harika bir film (Neill ve ark. 2012).

Farklı Gündemleri Yönetme

Her paydaşın farklı bir ihtiyaç gündemi vardır. Örneğin, işletme sahipleri paralarının karşılığını projelerden almanın yollarını ararlar. İş ortakları, bir sözleşme gibi oldukları için açık gereksinimler isterler. Üst yönetim, projelerden gerçekleştirilebilecek olandan daha fazla finansal kazanç beklemektedir. Ve sistemler ve yazılım geliştiricileri belirsizliği severler çünkü bu onlara çözümler geliştirme özgürlüğü verir.
Proje yöneticileri, teslim edilen üründeki düşük performansla ilgili yanlış suçlamalardan korunmak için gereksinimleri kullanabilir.

Aynı paydaş grubundaki kişiler arasında bile farklı gündemlerin neden var olabileceğini anlamanın bir yolu Rashomon etkisidir. Rashomon
Akira Kurosawa'nın yönettiği 1950 yapımı çok saygı duyulan bir Japon filmidir. Ana olay örgüsü, bir samurayın öldürülmesinin olaya tanık olan dört tanığın (samuray, karısı, bir haydut ve bir odun kesici) bakış açısıyla anlatılmasını içerir ve her birinin gizli bir ajandası vardır ve her birinin birbiriyle çelişen bir muhasebesini anlatır. olay. Kısaca, "bir olayı anlamanız, bakış açınız ve olayın sonucundaki ilgi alanlarınız gibi birçok faktörden etkilenir" (Lawrence 1996).

Akıllı gereksinim yöneticisi, önceden doğru soruları sorarak bu gündemleri yönetmeye çalışır. Andriole (1998), aşağıdaki soruların uygun olduğunu öne sürer:

1. Proje talebi nedir?
   a. Kim istiyor?
   B. İsteğe bağlı mı yoksa isteğe bağlı mı?
2. Projenin amacı nedir?
   a. Tamamlanırsa, yeni veya geliştirilmiş sistemin kurumsal performans üzerinde ne etkisi olacak?
   B. Kârlılık konusunda mı?
   C. Ürün geliştirme konusunda mı?
   D. Müşteri tutma ve müşteri hizmetleri hakkında?
3. İşlevsel gereksinimler nelerdir?
   a. Sistemin amacı yerine getirmek için yapması gereken belirli şeyler nelerdir? tam gereksinimler?
   B. Nasıl sıralanmalılar?
   C. Uygulama riskleri nelerdir?
4. Güvenlik, kullanılabilirlik ve birlikte çalışabilirlik gibi işlevsel olmayan gereksinimler nelerdir?
   a. Nasıl sıralanmalılar?

B. Uygulama riskleri nelerdir?
C. İşlevsel ve işlevsel olmayan gereksinimleri nasıl değiştirirsiniz?
5. Projeyi, işlevselliğini prototiplemek için yeterince iyi anlıyor muyuz?
6. Prototip kabul edilebilirse, herkes öncelikli işlevsellik ve teslim edilecek işlevsellik, ilk maliyet ve program tahminleri, tahminlerin yapısal belirsizliği, projenin kapsamı ve ek gereksinimlerin yönetimi konusunda imza atacak mı?

Andriole, bu soruları önceden sorarak gizli gündemlerin ortaya çıkarılabileceğini ve farklılıkların çözülebileceğini iddia ediyor. En azından, önemli konular önden gündeme getirilecek ve bu süreçte çok daha sonra değil.

Uzlaşma inşası

Aşağıdakiler dahil olmak üzere fikir birliği oluşturmaya yönelik çok sayıda yaklaşım vardır:

Müzakere (argüman)
Yönetici fiatı

Uzmanlara hitap ediyor

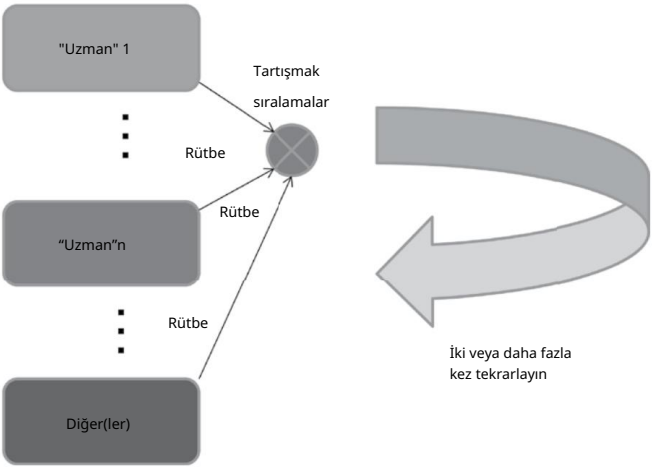
Örneğin, çeşitli paydaşların sıralamalarını ağırlıklandırmak için bulanık mantık kullanılarak matematiksel teknikler kullanılabilir. Daha yakın zamanlarda, Kassab (2014), gereksinimlerin zamanında efor tahmini için farklılıkları çözmede ve farklı görüşlerde tutarlılık bulmada yaygın olarak kullanılan analitik hiyerarşi sürecini (AHP) kullanmıştır. Ancak yaklaşımı, gereksinimlerin uyumlaştırılması ve müzakere edilmesi için kolayca uyarlanabilir.

Ancak sistem mühendisliğinde belki de en ünlü fikir birliği oluşturma tekniği Geniş Bant Delphi tekniğidir. 1970'lerde geliştirilen, ancak 1980'lerin başında Barry Boehm'de popüler hale gelen Wideband Delphi, genellikle alternatiflerin seçimi ve önceliklendirilmesi ile ilişkilendirilir. Bu alternatifler genellikle bir soru şeklinde sunulur:

Aşağıdaki skalaya göre tercihinizi değerlendirin (5=en çok arzu edilen, 4=istenen, 3=kararsız, 2=istenmeyen, 1=kiralamak istenen)

Alternatiflerin listesi ve ilgili ölçek, bu gereksinimleri sessizce ve bazen de anonim olarak sıralayan bir uzmanlar paneline (ve gereksinim sıralaması durumunda, paydaşlara) sunulur. Sıralama ölçeği herhangi bir sayıda seviyeye sahip olabilir. Toplanan sıralama listesi daha sonra bir koordinatör tarafından gruba yeniden sunulur. Genellikle, sıralamalarda önemli bir anlaşmazlık vardır. Bir tartışma yürütülür ve uzmanlardan görüş farklılıklarını gerekçelendirmeleri istenir.
Tartışmadan sonra bağımsız sıralama işlemi tekrarlanır (Şekil 9.1).

Her yinelemede, bireysel müşteri sıralamaları yakınsamaya başlamalı ve süreç, tatmin edici bir yakınsama düzeyine ulaşılana kadar devam eder.



Şekil 9.1 Geniş Bant Delphi işlemi.

Sürecin daha yapılandırılmış bir şekli:

1. Koordinatör, her uzmana bir spesifikasyon ve bir tahmin sunar form.
2. Koordinatör, uzmanların tahminleri tartıştığı bir grup toplantısı çağrısında bulunur koordinatör ve birbirleri ile ilgili sorunlar.
3. Uzmanlar formları isimsiz olarak doldurur.
4. Koordinatör, tahminlerin bir özetini hazırlar ve dağıtır.
5. Koordinatör, özellikle uzmanların tahminlerinin büyük ölçüde farklılık gösterdiği noktaları tartışmasına odaklanan bir grup toplantısı çağrısında bulunur.
6. Uzmanlar formları yine isimsiz olarak doldurur ve 4 ila 6 arasındaki adımlar şu şekilde tekrarlanır: uygun olarak birçok tur.

Geniş Bant Delphi sürecinde asla oybirliğiyle anlaşma olmayacak, ancak en azından katılan herkes onun görüşünün dikkate alındığını hissedecek. Geniş Bant Delphi, bir tür kazan-kazan müzakeresidir ve diğer karar verme türleri için kullanılabilir.

Beklenti Yeniden Ziyaret Edildi: Pascal'ın Bahsi

Matematikçi Blaise Pascal (1623-1662), Pascal Üçgeni (iki terimli katsayıları bulmanın uygun bir yolu) ve olasılıkta çalışma dahil olmak üzere çeşitli başarıları ile tanınır. Beklenen değer kavramına yol açan, olasılık teorisindeki çalışmasıydı ve daha sonraki yaşamında matematikten çok dinle ilgilenmeye başladığında böyle bir yaklaşımı kullandı. Onun manastır hayatı sırasında oldu

üstün bir varlığa inanıp inanmadığına bakılmaksızın, erdemli bir yaşam sürmenin tavsiye edildiğini öne süren bir teori geliştirdi. Beklenen değer teorisini kullanan yaklaşımı şimdi Pascal'ın bahsi olarak adlandırılıyor ve bu böyle gidiyor.

Bir kişinin bir inanç denemesi geçirdiğini ve bu yüce varlığa inanıp inanmadığından (buna tartışma uğruna Tanrı diyelim) inanıp inanmadığından emin olmadığını hayal edin. Pascal, böyle bir Tanrı'nın var olduğu (ya da olmadığı) nihai idrak karşısında, erdemli yaşamanın sonuçlarını değerlendirmenin değerli olduğunu öne sürer. Bunu görmek için Tablo 9.1'i inceleyin.

Tanrı'nın var olup olmadığının eşit derecede muhtemel olduğunu varsayarsak (bu büyük bir varsayım), erdemli yaşamanın beklenen sonucunun (sonucunun) cennetin yarısı, erdemsiz yaşamanın beklenen sonucunun ise lanetin yarısı olduğunu görürüz.
Bu nedenle, erdemli yaşamak kişinin yararınadır.

Pascal'ın bahsinin beklenti belirleme ile ne ilgisi var? Paydaşlar, çeşitli organizasyonel sorunları içeren ihtimalleri oynadıkları için bazen bilgileri saklayarak veya yetersiz bilgiler sunarak bahislerini hedge edeceklerdir. Örneğin, bir paydaş, kimsenin istemediğine inandığı ve alay konusu olabilecek bir özelliği talep etmek ister mi? Oyun teorisi açısından, onların fikirlerini saklamak daha güvenlidir. Bunu görmek için, Tablo 9.2'deki değiştirilmiş Pascal bahis sonuç matrisini inceleyin.

Paydaş belirli bir özellik hakkında (veya belirli bir özelliğin aksine) konuşmaya karar verirse, grubun aynı fikirde olması veya katılmaması ihtimalinin eşit olduğunu varsayarsa, sonuç matrisi, eğer grup isterse bir miktar övgü almayı bekleyebileceğini gösterir. kabul eder veya grup aynı fikirde değilse biraz alay eder. Ayrıca, karar vermede bireylerin, kazanma potansiyeline sahip olanlara kıyasla kayıptan kaçınan kararlar alma eğiliminde olacağı iyi bilinmektedir - çoğu birey riskten kaçınır.
Ayrıca, yüksek güç mesafelerine ve erkeklik endekslerine sahip ülkelerden gelen paydaşlar için olası kültürel etkileri hatırlayın. Yukarıdaki analiz aynı zamanda anlaşma ve anlaşmazlık olasılıklarının

Tablo 9.1 Pascal'ın Bahis Sonuç Matrisi

	Tanrı vardır	Tanrı yok
erdemli yaşa	cennete ulaşmak	Boş
erdemli yaşama	Lanet olsun	Boş

Tablo 9.2 Değiştirilmiş Pascal'ın Bahis Sonuç Matrisi

	Grup Kabul Ediyor	Grup Katılmıyor
Sesli söyle	övgü almak	alay konusu olmak
Sessiz kalır	Hiçbir şey olmuyor	Hiçbir şey olmuyor

aynı—paydaş, meslektaşlarının aynı fikirde olmayacağına dair güçlü bir şans olduğuna inanıyorsa, beklenen sonuçlar çok daha kötüdür. Tabii ki, daha sonraki süreçte bu özellik aniden başkaları tarafından önemli olarak keşfedilebilir. Ancak artık oyunda çok geç oldu ve bu özelliği eklemek maliyetli. Paydaş ancak başlangıçta, tartışma için güvenli bir ortamda konuşsaydı, büyük bir maliyet ve sorundan kaçınılabilirdi.

Pascal'ın bahsi, beklentisi ve riski üzerine son bir yorum. Yazar bir zamanlar, tüm yeni çalışanları hoş bir öğle yemeği ile karşılayan bir patron için çalıştı - ona Bob diyeceğiz. O öğle yemeğinde "Ben 'sürpriz olmayan' bir adamım. Sen beni şaşırtma, ben de seni şaşırtmayacağım. Bu yüzden, aklınıza takılan herhangi bir şey olursa ya da herhangi bir sorun ortaya çıkarsa, bunu hemen dikkatime getirmenizi istiyorum." Bu duygu kulağa harika geliyordu. Bununla birlikte, yazar ya da başka biri Bob'a kötü haber getirdiğinde, durumdan haberci sorumlu olsun ya da olmasın, Bob yığınını patlatır ve daha iyi olanı yapan talihsizleri azarlardı. Bir süre sonra, potansiyel haberciler Bob'a bilgi getirmekten vazgeçeceklerdi. Karar tamamen oyun teorisiydi - eğer kötü bir haber alırsan ve bunu Bob'a getirirsen, sana bağırırdı. Eğer o bunu öğrenmediyse (sıklıkla olduğu gibi çünkü başka kimse de ona söylemezdi), o zaman onun koçluk sayfasından kaçtınız. Kötü haberi bir şekilde öğrenirse, size bağırılabilir - ancak sorundan sorumlu taraf siz olsanız bile, size değil, gördüğü ilk kişiye bağırabilir. Bu yüzden susmak mantıklıydı (ve oyun teorisine göre kesinlikle sağlamdı).

"Sürpriz değil Bob" un her zaman şaşırması oldukça ironikti çünkü herkes ona bir şey söylemekten korkuyordu. Buradan alınacak ders şu ki, haberciyi vurursanız, insanlar size bilgi getirmenin sonuçlarını anlamaya başlayacak ve yakında o bilgiden mahrum kalacaksınız. Proje yaşam döngüsü boyunca gereksinim bilgilerini aktif olarak aramak ve davet etmek, gereksinim yönetiminin önemli bir yönüdür.

Küresel Gereksinim Yönetimi

Gereksinim mühendisliği, yazılım geliştirmede işbirliğinin en yoğun olduğu faaliyetlerden biridir. Yetersiz sosyal iletişim, coğrafi olarak dağılmış paydaşlar ve gereksinimler nedeniyle, mühendisler uygun araçlar olmadan gereksinimleri anlamakta güçlük çekerler. Çevik geliştirme, bu sorunu her zaman yerinde müşteriler aracılığıyla çözmeyi amaçlar (Sinha ve Sengupta 2006).

Küresel ve hatta karada dış kaynak kullanımı, gereksinim mühendisliği çabalarına her türlü zorluğu sunar. Bunlar arasında zaman gecikmeleri ve saat dilimi sorunları, gerektiğinde müşteri ve satıcı sitelerine fiziksel seyahatin maliyetleri ve stresleri ve sanal konferans ve telefonun dezavantajları yer alır. Küresel olarak dağıtılan birçok proje için gayri resmi e-postalar ve e-posta ile dağıtılan belgeler ana form olmasına rağmen, basit e-posta iletişimlerine bile tamamen güver

işbirliği. Ancak e-posta kullanımı, sık sık bağlam değiştirmeye, bilgi parçalanmasına ve sözel olmayan ipuçlarının kaybolmasına yol açar.

Offshoring, farklı bir ana dil ve önemli ölçüde farklı kültüre sahip bir ülkede gerçekleştiğinde, çalışma programları, çalışma tutumları, iletişim engelleri ve işin nasıl yürütüleceğine ilişkin müşteri-satıcı beklentileri açısından yeni sorunlar ortaya çıkabilir (Hofstede'nin metriklerini hatırlayın) . Ayrıca, offshoring yeni bir risk faktörü sunar: jeopolitik risk ve bu risk anlaşılmalı, nicelleştirilmeli ve bir şekilde gereksinim mühendisliği süreci ve programına dahil edilmelidir. Son olarak, dünya çapında yasalarda, yasal süreçlerde ve hatta ticari işlemlerde dürüstlük beklentilerinde büyük farklılıklar vardır. Bu konular özellikle gereksinimlerin ortaya çıkarılması aşamasında önemlidir.

Bhat et al. (2006) gözlemledikleri dokuz özel sorunu vurguladılar veya Tecrübeli. Bunlar dahil:

- Çakışan müşteri-satıcı hedefleri
- Düşük müşteri katılımı
- Çatışan gereksinim mühendisliği yaklaşımları (müşteri ve satıcı arasında)
- Müşteri taahhüdünün proje hedefleriyle yanlış hizalanması
- Takım seçiminde anlaşmazlıklar
- İletişim sorunları
- Sorumluluğu reddetmek
- Oturum kapatma sorunları
- Beklentilerle yanlış hizalanmış araçlar

Bhat et al. proje deneyimlerinin bir analizine dayanarak, bu durumlarda aşağıdaki başarı faktörlerinin eksik olduğunu öne sürüyorlar:

- Paylaşılan hedef, yani bir proje metaforu
- Paylaşılan kültür—sosyolojik anlamda değil, proje anlamında
- Paylaşılan süreç
- Paylaşılan sorumluluk
- Güven

Bu öneriler, gereksinim mühendisliğine yönelik çevik yaklaşımları kullanmanın zorluklarını ve avantajlarını daha önce tartışmış olsak da, çevik metodolojilerle büyük ölçüde tutarlıdır.

Son olarak, küresel olarak dağıtılmış gereksinim mühendisliği sürecinde araçlar nasıl bir rol oynayabilir? Sinha ve Sengupta (2006), bu amaç için pek çok uygun araç olmamasına rağmen yazılım araçlarının önemli bir rol oynayabileceğini öne sürmektedir.
Uygun yazılım araçları şunları desteklemelidir:

- Gayri resmi işbirliği
- Yönetimi değiştir

Farkındalığı teşvik etme (örneğin, tetikleyiciler meydana geldiğinde paydaşlara otomatik e-posta gönderme)
Bilgi yönetimi—yapılandırılmamış proje bilgilerini kaydetmek ve ilişkilendirmek için bir çerçeve sağlamak

olduğunu iddia eden birkaç ticari ve hatta açık kaynaklı çözüm vardır.
bu özellikleri sağlamaktayız, ancak bunların ürün araştırmasını okuyucuya bırakıyoruz.

Gereksinim Yönetiminde Anti Modeller*

Sorunlu organizasyonlarda, başarının önündeki en büyük engel, sıklıkla problemin doğru tanımlanmasıdır. Organizasyonel işlev bozukluğunu teşhis etmek, gereksinim mühendisliği problemlerine yol açacak temel problemlerle uğraşırken oldukça önemlidir.

Tersine, sorunlar doğru bir şekilde tanımlandığında, hemen hemen her zaman uygun şekilde ele alınabilirler. Ancak örgütsel atalet çoğu zaman durumu gölgeler veya doğru şeyi yapmaktansa yanlış şeyi yapmayı kolaylaştırır. Sorunu yanlış anladıysan, doğru olanın ne olduğunu nasıl bilebilirsin?

Çığır açan kitaplarında Brown ve ark. (1998), yazılım mimarisinde ve tasarımında ve yazılım projelerinin yönetiminde ortaya çıkabilecek problemlerin veya anti modellerin bir sınıflandırmasını tanımladı. Ayrıca bu durumlar için çözümler veya yeniden düzenlemeleri tanımladılar. Böyle bir sınıflandırma sağlamanın yararı, sorun durumlarının hızlı ve doğru bir şekilde tanımlanmasına yardımcı olması, sorunları ele almak için bir oyun kitabı sağlaması ve bu durumlarda kuşatılmış çalışanlara, gerçekte teselli bulabilecekleri için biraz rahatlama sağlamasıdır. yalnız olmadıklarını.

Bu anti-kalıplar, bireysel yöneticiden örgütsel işlev bozukluğu yoluyla ortaya çıkar ve kötü ifade edilmiş, eksik, yanlış veya kasıtlı olarak bozucu gereksinimlerde tezahür edebilir.

Antipattern setimiz, 28 çevresel (organizasyonel) ve 21 yönetim antipatterninden oluşan neredeyse eşit bir bölünmeden oluşur. Yönetim karşıt kalıplarına, bireysel bir yönetici veya yönetim ekibi ("yönetim") neden olur. Bu karşıt modeller, bir gruba, departmana veya organizasyona liderlik edecek yetenek veya mizaçtan yoksun denetçilerdeki sorunları ele alır. Çevresel anti-kalıplar, hakim bir kültür veya sosyal modelden kaynaklanır. Bu karşıt modeller, yanlış yönlendirilmiş şirket stratejisinin veya kontrolsüz sosyo-politik güçlerin sonucudur. Ancak, özellikle gereksinim mühendisliğinde uygulanabilir olan antipatterns setinin yalnızca küçük bir alt kümesini tanımlamayı seçiyoruz.

* Bu tartışmanın bir kısmı Neill ve ark. (2012) izni ile.

Çevresel Antikalıplar

Farklı Hedefler

Herkes aynı yöne çekmeli. İşletmenin amaçlarıyla uyuşmayan bireysel veya gizli ajandalara yer yoktur. Farklı yönlere çekenler olduğunda farklı hedefler antipattern var olur.

Farklı hedeflerle ilgili doğrudan ve dolaylı birçok sorun vardır.

Bir kuruluşun misyonuna aykırı olan gizli ve kişisel gündemler, kaynakları stratejik olarak önemli görevlerden mahrum bırakır.

Örgütler, kendi tanıtımlarını yapmak için klikler oluştukça parçalanır kişisel çıkarlar.

Personel, fermanlar ve değişiklikler için gerçek nedenleri deşifre etmeye çalıştığından, kararlar ikinci bir tahmine tabidir ve "tekrar yetkilisinin incelemesine" tabidir.

Stratejik hedefler, herkes onlara doğru çalışırken ulaşılması yeterince zordur, tam destek olmadan imkansız hale gelir ve kuruluş için risk oluşturur.

Paydaş uyumsuzluğu ile farklı hedefler arasında güçlü bir uyum vardır, bu nedenle her ikisinin de varlığının çok iyi farkında olun.

Farklı hedefler kazara ve kasıtlı olarak ortaya çıkabileceğinden, iki grup vardır. çözümler veya yeniden düzenleme.

İlk anlama ve iletişim sorunuyla uğraşmak, günlük kararların daha büyük hedefler üzerindeki etkisini açıklamayı içerir. Bu, Bölüm 2'de açıklanan kutu şirketinin kurumsal yöneticileri için geçerliydi.

Yöneticiler büyük resmi unuttular. Bununla birlikte, misyon beyanıyla birlikte bir kahve kupası sağlamaktan daha fazlası var. Yanlış anlamanın, personelin misyon veya hedeflerin farkında olmamasından kaynaklanmadığını unutmayın; kuruluşlar genellikle bunları yaymada çok iyidir. Kararlarının bu hedefler üzerinde herhangi bir etkisi olduğunu anlamıyorlar. Örgüte çok dar bir bakış açısına sahipler ve bu bakış açısı genişletilmelidir.

Ancak, kasıtlı olarak karşıt bir rota çizmenin ikinci sorunu çok daha sinsidir ve önemli ölçüde müdahale ve gözetim gerektirir.

Başlangıç noktası, onların kişisel hedefleri ile organizasyonun hedefleri arasındaki kopukluğun farkına varmaktır. Neden belirtilen hedeflerin yanlış olduğunu düşünüyorlar? Motifler gerçekten kişiselse, kişisel başarılarının organizasyonun başarısı ile gelmeyeceğini düşünüyorlarsa, radikal değişikliklere ihtiyaç vardır. Aksi takdirde en iyi yol, onların örgütsel hedeflere katılmalarını sağlamaktır. Bu, her paydaşın temel misyon ve hedeflerin tanımında ve yayılmasında temsil edilmesi ve ardından bilgilendirilmesi, güncellenmesi ve temsil edilmesi durumunda en kolay şekilde elde edilir.

Süreç Çatışması

Süreç çatışması, kanıtlanmış bir hibrit süreç tanımlanmadan farklı süreçlerin savunucularının birlikte çalışması gerektiğinde ortaya çıkabilecek sürtüşmedir. İşlevsizlik, kuruluşların iki veya daha fazla iyi niyetli ancak tamamlayıcı olmayan süreçleri olduğunda ortaya çıkar; katılanlar için büyük bir rahatsızlık yaratılabilir.
İşlevsel gruplar veya şirketler (farklı süreçlere sahip) birleştiğinde veya yönetim aniden eski bir süreci değiştirmek için yeni bir süreç başlatmaya karar verdiğinde süreç çatışması ortaya çıkabilir.
Bu karşıt kalıbın belirtileri arasında zayıf iletişim yer alır—
hatta düşmanlık—yüksek devir ve düşük üretkenlik.

Bir süreç çatışmasının çözümü, hibritleştirilmiş bir yaklaşım geliştirmeyi içerir—
süreçlerin arayüzlerindeki farklılıkları çözen bir tane. Yeniden eğitim ve çapraz eğitim de kullanılabilir. Örneğin, tüm mühendislik gruplarını gereksinim mühendisliği ilke ve uygulamaları konusunda eğiterek, karşılıklı anlayış sağlanabilir. Başka bir çözüm, çatışmayı çözen üçüncü bir sürece geçmektir.

Yönetim Anti Modelleri

Metrik Kötüye Kullanım

Gereksinim mühendisliğinde ortaya çıkabilecek ilk yönetim karşıt modeli, metrik kötüye kullanımıdır, yani metriklerin yetersizlik veya kasıtlı kötülük yoluyla kötüye kullanılmasıdır (Dekkers ve McQuaid 2002).

Birçok süreç iyileştirme çabasının merkezinde bir ölçüm programının tanıtılması yer alır. Aslında bazen ölçüm programı süreç iyileştirmedir. Başka bir deyişle, bazı insanlar, ölçmenin yönetimde oynadığı rolü yanlış anlıyor ve onun salt varlığını bir gelişme olarak yanlış yorumluyor.

Bu doğru bir varsayım değil. Metrikte kullanılan veriler yanlışsa veya metrik yanlış şeyi ölçüyorsa, bunlara dayalı olarak verilen kararlar muhtemelen yanlıştır ve yarardan çok zarar verir.

Tabii ki, metrik kötüye kullanımından kaynaklanabilecek önemli sorunlar, sorunun kökenine bağlıdır: yetersizlik veya kötülük. Yetersiz metrik kötüye kullanımı, nedensellik ve korelasyon arasındaki farkı anlayamamaktan kaynaklanır; dolaylı önlemlerin yanlış yorumlanması; Bir ölçüm programının etkisinin hafife alınması. İşte böyle bir sorunun kaynağına bir örnek. Bir yangın durumunda yangın geciktirici dağıtmak için bir fabrika için bir yangın kontrol sisteminin gerekli olduğunu varsayalım.
Yangın, sıcaklığa, dumanın varlığına, oksijenin yokluğuna ve yanmadan kaynaklanan gazların varlığına vb. bağlı olarak çeşitli şekillerde tespit edilebilir.
Peki, yangın olup olmadığını belirlemek için bunlardan hangisi ölçülmelidir? Yanlış metriği seçmek, metriklerin kötüye kullanılmasına neden olabilir.

Kötü amaçlı metriklerin kötüye kullanılması, kişisel bir gündeme dayalı olarak belirli bir konumu destekleyen veya kötüleyen metriklerin seçilmesinden kaynaklanır. Örneğin, bir yönetici varsayalım.

günlük olarak mühendis başına yazılan gereksinimlerin sayısını izleyen bir politika oluşturur ve bu ölçü etrafında bir tazminat algoritması oluşturur. Böyle bir yaklaşım basittir ve farklı türden gereksinimleri ortaya çıkarma, analiz etme, kabul etme ve yazma konusundaki değişen zorlukları hesaba katmaz. Aslında, politika tamamen titizlikle çalışan, ancak yöneticinin tercihi için çok yavaş çalışan bir kişiyi seçmek için oluşturulmuş olabilir.

Metrik kötüye kullanımı için çözüm veya yeniden düzenleme, rahatsız edici ölçümleri durdurmaktır. Hiçbir şeyi ölçmemek, yanlış şeyi ölçmekten daha iyidir. Veriler mevcut olduğunda, insanlar, doğruluklarına bakılmaksızın, onları karar vermede kullanırlar.

Güverteler temizlendikten sonra Dekkers ve McQuaid, anlamlı bir ölçüm programının başlatılması için gerekli bir dizi adım önerir.
(Dekkers ve McQuaid 2002):

1. Ölçüm amaçlarını ve planlarını tanımlayın—belki de hedef soru metriği (GQM) paradigmasını uygulayarak.
2. Ölçümü sürecin bir parçası haline getirin— ona, bütçesinde kesintiye gidebilecek veya bir gün tamamlamayı umduğunuz başka bir proje gibi davranmayın.
3. Kapsamlı bir ölçüm anlayışı kazanın— doğrudan ve dolaylı metrikleri, nedenselliğe karşı korelasyonu ve en önemlisi, metriklerin yorumlanması ve buna göre hareket edilmesi gerektiğini anladığınızdan emin olun.
4. Kültürel konulara odaklanın—bir ölçüm programı kuruluşun kültürünü etkileyecektir; onu bekle ve bunun için plan yap.
5. Gerçek verileri toplamak ve raporlamak için güvenli bir ortam yaratın —iyi bir gerekçe olmadan insanların yeni ölçütlerden şüphe duyacaklarını ve koyun giysilerinde bir zaman ve hareket çalışmasından korkacaklarını unutmayın.
6. Değişime yatkınlık geliştirin— ölçüler eksiklikleri ve verimsizlikleri ortaya çıkaracak, bu nedenle iyileştirmeler yapmaya hazır olun.
7. Tamamlayıcı bir önlemler paketi geliştirin —tek bir metriğe ayrı ayrı yanıt vermenin olumsuz yan etkileri olabilir. Bir dizi ölçüm bu riski azaltır.

Metriklerin yanlış yönetildiğine inanıyorsanız, metriklerin neden toplandığını, nasıl kullanıldığını ve bu tür bir kullanım için herhangi bir gerekçe olup olmadığını yönetime sorgulayarak yukarıdaki süreci başlatmaya çalışabilirsiniz. Ayrıca, alternatif metrikler ve mevcut metriklerin uygun kullanımı veya daha uygun kullanımları fırsatlarıyla metriklerin düzeltici bir şekilde anlaşılmasını sağlamayı da teklif edebilirsiniz.

Mantar Yönetimi

Mantar yönetimi, yönetimin personel ile etkili bir şekilde iletişim kuramadığı bir durumdur. Esasen, herkesi “şişman, aptal ve mutlu” tutmak için bilgi kasıtlı olarak saklanmaktadır. Adı gerçeğinden türetilmiştir

mantarlar karanlıkta ve loş ışıkta gelişir ama güneş ışığında ölür. Eskilerin dediği gibi, "Onları karanlıkta tutun, gübre ile besleyin, büyümelerini izleyin... ve işiniz bittiğinde kafalarını kesin."

İşlev bozukluğu, ekip üyeleri büyük resmi gerçekten anlamadığında ortaya çıkar; Etkiler, özellikle gereksinim mühendisliği açısından, paydaşlar dışarıda bırakıldığında önemli olabilir. Ön saflarda çalışan birinin büyük resmi anlamaya ihtiyacı olmadığını varsaymak biraz aşağılayıcı. Ayrıca, örneğin doğrudan müşterilerle çalışanlar, şirket üzerinde kapsamlı bir etkisi olabilecek mükemmel fikirlere sahip olabilir.

Bu nedenle mantar yönetimi, düşük çalışan moraline, işten ayrılmaya, kaçırılan fırsatlara ve genel başarısızlığa neden olabilir.

Mantar yönetimini sürdürmek isteyenler, bilgi, strateji ve verileri açıklamamak için mazeretler bulacaktır. Bu durumu yeniden düzenlemek için bazı basit stratejiler şunları içerir:

Daha fazla şeffaflık talep etmenizi sağlayan sorunları sahiplenin.
Kendi başınıza bilgi arayın. Dışarıda. Onu bulmak için daha çok çalışmanız gerekiyor ve parçaları bir araya getirmeniz gerekebilir. Siz ve diğer mantarlar arasında, büyük resmin çoğunu görebilirsiniz.
Açık kitap yönetimi kültürüne dönüşümü savunun.

Tüm yeniden düzenlemeyle birlikte, değişimi etkilemek için cesaret ve sabır gerekir.

Gereksinim Yönetimi için Diğer Paradigmalar

Gereksinim Yönetimi ve Doğaçlama Komedisi

Doğaçlama komedi, işbirliği (ve sonuçta gereksinim mühendisliği, işbirliğinde en üst noktadır) ve gereksinim yönetiminde zorluklarla başa çıkmak için bazı teknikler sağlayabilir. Doğaçlama komediden (örneğin, Whose Line Is It Neyse?) hoşlanan herkes, çok farklı bakış açılarına sahip kişilerin mükemmel etkileşimini ve bu farklılıkların çözümünü görmüştür. Yazar doğaçlama komedi okudu ve bu sanattan alınabilecek birçok dersi gözlemledi.

Dinleme becerileri gerçekten önemlidir - hem müşterilerin ve diğer paydaşların söylediklerini duymak hem de gereksinim mühendisliği çabasında ortaklarınızı/ortaklarınızı oynamak için.
Anlaşmazlık veya kısmi anlaşma olduğunda en iyi yanıt "evet, ama..." yerine "evet ve..." yani fikirleri yıkmak yerine üzerine inşa etmektir.
İşler ters gidecek - hem doğaçlama komedi hem de gereksinim mühendisliği uyum sağlamakla ilgilidir.

Zorluklar karşısında eğlenmelisiniz.
Son olarak, yalnızca kontrol edilebilir olanı kontrol ederek tepki vermeyi öğrenmelisiniz (genellikle, olayların kendilerine değil, belirli olaylara yalnızca kendi tepkinizdir).

Dinleme becerilerinizi, duygusal zekanızı ve ayaklarınızın üzerinde düşünme yeteneğinizi geliştirmenize yardımcı olması için doğaçlama komediden bazı teknikleri gerçekten uygulayabilirsiniz; bu da bir gereksinim mühendisi ve bir takım oyuncusu olarak uygulamanızı geliştirecektir.

Örneğin, "Zip, Zap, Zup (veya Zot)" adlı doğaçlama bir beceri geliştirme oyununu düşünün.* İşte nasıl çalıştığı. Dört veya daha fazla kişiyi (ne kadar çok olursa o kadar iyi) yüzü içe dönük bir daire oluşturacak şekilde düzenleyin. Bir kişi zip, zap veya zup diyerek başlar. Bu kişi size bakarsa, dairedeki başka birine bakar ve sırayla yanıt verirsiniz: zip, zap veya zup. Katılımcıların üç kelimeden başka sözlü iletişime izin verilmez. Oyun, tüm katılımcılar üç yanıttan hangisinin verileceğini tahmin etmeye başlayana kadar devam eder. Oyunu oynamak göründüğünden çok daha zordur ve tepkileri tahmin etme yeteneği (eğer elde edilirse) birkaç dakika sürebilir. Bu oyunun/egzersizin amacı, katılımcıları duyguları "duymaya" ve diğer sözsüz ipuçlarını yakalamaya zorlamasıdır.

Başka bir oyunda, Dr. Her şeyi bilen, üç veya daha fazla katılımcıdan oluşan bir grup, her katılımcının her seferinde yanıtın yalnızca bir kelimesini vererek soruları birlikte yanıtlar. Bu nedenle, bir gereksinim mühendisliği çalışmasında, paydaş grubu A'dan bir katılımcı koleksiyonu toplayacak ve onlara aşağıdaki soruyu soracağız. "Lütfen aşağıdaki cümleyi tamamlayınız: sistem aşağıdakileri sağlamalıdır..." daha sonra katılımcılar sırayla birer kelime cevabını verirler. Bu çok zor bir deneyimdir ve bir gereksinim belirleme tekniği olarak tasarlanmamıştır - bu bir düşünce oluşturma ve ekip oluşturma alıştırmasıdır ve gereksinim mühendisini ve katılımcıları gelecekteki zorluklar hakkında bilgilendirmeye yardımcı olabilir.

Son bir alıştırma, gereksinim mühendisinin aynı anda iki kişiden gelen soruları yanıtlamasını içerir. Bu deneyim, gereksinim mühendisinin kendi ayakları üzerinde düşünmesine yardımcı olur ve aynı zamanda iki farklı paydaştan/müşteriden gelen sorulara aynı anda yanıt vermeleri gerekebilecek müşterilerle etkileşime girerken sıklıkla yaşayacaklarını simüle eder.

Bu yazar, gereksinimlerin ortaya çıkarılması için uygun tür olarak komediyi tercih ederken, diğerleri (örneğin, Mahaux ve Maiden 2008) gereksinimleri keşfetme için bir paradigma olarak doğaçlama tiyatroyu incelediler. Her halükarda, beynimizin en iyi fikirlerimizi bastırma eğiliminde olduğu ve doğaçlamanın kendiliğinden ve cömertçe düşünmenize yardımcı olduğu açık görünüyor. Bu nedenle, bu egzersizleri eğlenmek ve bu önemli becerileri geliştirmek için deneyin.

* Yazar bunun da geleneksel bir içme oyunu olduğunu kabul ediyor.

Senaryo Yazımı Olarak Gereksinim Yönetimi

Film senaryolarının (senaryoların) yazılmasının birçok yönden gereksinim mühendisliği ile bazı benzerlikleri vardır. Örneğin Norden (2007), gereksinim mühendislerinin senaryo yazma sürecini gözlemleyerek farklı bakış açılarını nasıl çözebileceklerini nasıl öğrenebileceklerini açıklar. Film yapımı ve yazılım arasında başka benzerlikler de var. Bunlar şunları içerir:

Filmler, film geliştikçe karşılanamayacak belirli beklentiler oluşturulurken önceden duyurulur (örneğin, oyunculardaki, senaristlerdeki, yönetmenlerdeki, olaylardaki ve çıkış tarihindeki değişiklikler). Yazılımlar ve sistemler genellikle gerçek sürümlerinden önce duyurulur ve daha sonra duyurulan işlevsellik ve sürüm tarihlerindeki değişiklikler ile birlikte duyurulur.

Egolar genellikle filmlerde, yazılımlarda ve sistemlerde önemli bir faktördür.

Film ve yazılım yapımında genellikle çok fazla el vardır ve sistemler.

Bazen filmlerin ihtiyaçları teknolojiyi aşar (ve bunun sonucunda yeni teknolojilerin geliştirilmesi gerekir). Aynı şey yazılım ve sistemler için de geçerlidir.

Filmler genellikle bütçe ve dağıtım beklentilerini aşıyor. Yazılım hakkında daha fazla şey söylememize gerek var mı?

Filmler sıra dışı olarak çekilir, gereksinimler belirtilir ve yazılım ve bazı sistemler de bu şekilde kurulabilir.

Filmler sırayla çekilir ve daha sonra anlamlı olması için birleştirilir. Yazılım da neredeyse her zaman bu şekilde geliştirilir. Bazı sistemler de öyle.

Pek çok iş sonunda çöpe atılıyor—filmlerde film kesim odasının zemininde bırakılmış olarak son buluyor; yazılımda, atılabilir prototipler ve tek kullanımlık araçlar olarak. Sistem bileşenleri genellikle atılmak üzere yapılmasa da, birçok test armatürü öyledir.

Büyük resim üretiminden ne öğrenebiliriz? Norden'ın makalesindeki kısa bir skeçte Jones, senaryo yazımından gereksinim mühendisleri için aşağıdaki ipuçlarını sağlar:

Hazırlık her şeydir. Doğrudan ayrıntıya atlamayın, ancak işinizi yapın. önce ev ödevi.

Birçok taslak üzerinde çalışmayı bekleyin. Farklı sürümlerin her birinin planlamayı desteklemesi ve paydaşlar arasında işbirliğini sağlaması gerekebileceğini unutmayın.

Gelecekteki sistem kullanıcıları hakkında ayrıntılı düşünün - geçmişleri, tutumları, alışkanlıklar, hoşlandığı ve hoşlanmadığı şeyler.

Yalnızca kullanıcıların fiziksel eylemlerini düşünmeyin. Onların ne olduğunu hatırla düşünebilir (bilişsel eylemleri) ve hissedebilir.

Gereksinimlerinizin hikayesini hatırlayın. Gereksinimler, kullanıcı hedeflerine ulaşmanın tam hikayesini anlatmalıdır.
Hedef kitlenizi tutun—birinin gereksinimler belgenizi okuması gerektiğini unutmayın. Belki de gerilime ve dramatik ironiye yer vardır (Norden 2007).

Gereksinim Yönetimi Standartları*

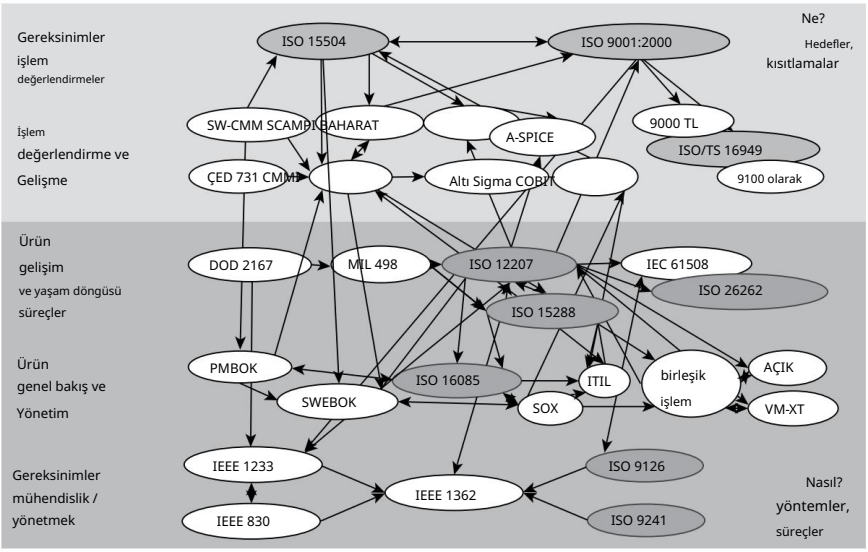
Gereksinim mühendisliği çabalarını bilgilendirmek için kullanılabilecek referans süreç modelleri, yönetim felsefeleri veya kalite standartları sağlayan birçok farklı uluslararası standart vardır. Bu standartlar, tamamlayıcı bir şekilde kullanılabilecekleri için birbirini dışlamaz. Şekil 9.2, ilgili yazılım mühendisliği ile ilgili standartları ve bunların nasıl ilişkili olduğunu özetlemektedir.

Şekil 9.2'de ISO standartları gölgelendirilmiştir ve ilgili standartlar aşağıdaki kategorilerde gruplandırılmıştır:

1. ISO 15504 (yetenek olgunluğunu belirleme ve süreç değerlendirme standardı) ve ISO 9001 (kalite yönetimi) gibi iyileştirme standartları (metamodeller).
2. SCAMPI (süreç iyileştirme için standart CMMI değerlendirme yöntemi) ile CMMI (yetenek olgunluk modeli entegrasyonu) gibi kalite yönetimi ve iyileştirme çerçeveleri; SPICE (yazılım süreci iyileştirme ve yetenek belirleme); ISO 15504; BT organizasyonları için COBIT (bilgi ve ilgili teknoloji için kontrol hedefleri); Altı Sigma; Telekomünikasyon kalite yönetim sistemlerine ilişkin gereksinimleri ve ölçümleri detaylandıran TL 9000; Otomotiv sistemlerinde kalite yönetimi gerekliliklerini sağlayan ISO/TS (teknik standart) 16949; ve havacılık endüstrisi için özel gereksinimlerle ISO 9001'i geliştiren AS (havacılık uzay standardı) 9100.

3. Süreç ve yaşam döngüsü modelleri DoD'yi (ABD Savunma Bakanlığı) içerir 2167 MIL (askeri standart) 498, ISO 15288 (sistem yaşam döngüsü) ve ISO 12207 (yazılım yaşam döngüsü).
4. Süreç uygulaması ve yönetişim düzenlemeleri ve politikaları arasında Sarbanes–Oxley Yasası (SOX), ISO 16085 (risk yönetimi) gibi standartlar, SWEBOK ve PMBOK (proje yönetimi bilgi gövdesi) gibi fiili standartlar ve çeşitli çerçeveler (Eber 2010).

* Bu bölüm, izin alınarak Laplante (2006) ve Ebert'ten (2010) alıntılanmıştır.



Şekil 9.2 Gereksinim mühendisliğine uygulanabilir standartlar. (Ebert, C.'den, Gereksinimler mühendisliği: Yönetim. P. Laplante'de (Ed.), Yazılım Mühendisliği Ansiklopedisi, pp. 932–948, Taylor & Francis, 2010. İzinle.)

SWEBOK'u Bölüm 1'de zaten ele almıştık ve IEEE Standard 830, Bölüm 4 ve 5'te tartışılan IEEE Standard 29148 kapsamındaydı.
Aşağıdaki bölümlerde diğer standartlardan bazılarını inceleyeceğiz.

Yetenek Olgunluk Modeli Entegrasyonu

Yetenek olgunluk modeli entegrasyonu (CMMI), beş seviyeden oluşan bir sistem ve yazılım kalite modelidir. CMMI bir yaşam döngüsü modeli değil, süreç olgunluğunun altında yatan ilkeleri ve uygulamaları tanımlayan bir sistemdir. CMMI, yazılım organizasyonlarının, ad hoc, kaotik süreçlerden olgun, disiplinli süreçlere evrimsel bir yol açısından süreçlerinin olgunluğunu iyileştirmelerine yardımcı olmayı amaçlamaktadır.

Carnegie Mellon Üniversitesi'ndeki Yazılım Mühendisliği Enstitüsü tarafından geliştirilen CMMI, beş olgunluk düzeyinde düzenlenmiştir. Bir organizasyonun yazılım süreçlerinin öngörülebilirliği, etkinliği ve kontrolünün, organizasyon bu beş seviyeyi yukarı çıktıkça iyileştirdiğine inanılmaktadır. Gerçekten kesin olmasa da, bu pozisyonu destekleyen bazı ampirik kanıtlar var.

Yetenek olgunluk modeli entegrasyonu, hem yüksek seviyeli hem de düşük seviyeli gereksinim yönetimi süreçlerini tanımlar. Üst düzey süreçler yöneticiler ve ekip liderleri için geçerliyken, düşük düzeyli süreçler analistler, tasarımcılar, geliştiriciler ve test uzmanları için geçerlidir.

Tipik üst düzey gereksinim uygulamaları/süreçleri şunları içerir:

- Organizasyonel politikalara bağlı kalmak
- Belgelenmiş proje planlarının takibi
- Yeterli kaynakların tahsis edilmesi
- Sorumluluk ve yetki atama
- Uygun personeli eğitmek
- Tüm öğeleri sürüm veya konfigürasyon kontrolü altına almak ve tüm (mevcut) paydaşlar tarafından gözden geçirilmek
- İlgili standartlara uymak
- Üst yönetimle durumu gözden geçirme

Düşük seviyeli en iyi uygulamalar/süreçler şunları içerir:

- Gereksinimleri anlama
- Tüm katılımcıların gereksinimleri karşılamasını sağlamak
- Yaşam döngüsü boyunca gereksinim değişikliklerini yönetme
- Gereksinim izlenebilirliğini yönetme (ileriye ve geriye doğru)
- Proje planları ile arasındaki tutarsızlıkların belirlenmesi ve düzeltilmesi Gereksinimler

Bu uygulamalar, metin boyunca tartışılanlarla tutarlıdır.

CMMI için 3. seviye ve daha yüksek seviyelere ulaşmak, bu en iyi uygulamaların bir organizasyon içinde belgelenmesini ve takip edilmesini gerektirir.

CMM kalite modelleri ailesi, gereksinim yönetimini şu şekilde tanımlar:

> yazılım projesinin gereksinimleri konusunda müşteri ile bir anlaşmanın oluşturulması ve sürdürülmesi. Anlaşma, yazılım yaşam döngüsü boyunca yazılım projesinin faaliyetlerini tahmin etmek, planlamak, gerçekleştirmek ve izlemek için temel oluşturur. (Paulk ve diğerleri 1993)

Maksimum esneklik için CMM, hangi araçları veya teknikleri belirlemez
Bu hedeflere ulaşmak için kullanmak için.

ISO 9001

ISO Standardı 9000, Kalite Yönetimi, kalite iyileştirme için genel, dünya çapında bir standarttır. ISO 9000, ISO 9001 (Kalite Yönetim Sistemleri—Gereksinimler), ISO 9004 (Bir Kuruluşun Sürdürülebilir Başarısını Yönetme) ve ISO 9011 (Yönetim Sistemleri Denetimi için Kılavuz İlkeler) olmak üzere dört ciltte topluca açıklanan standart, çok çeşitli ortamlarda uygulanabilir. ISO 9000 kapsamına göre işletmeler için geçerlidir.

onların faaliyetleri. Bu ISO standartları, şirketlerin kaliteli bir ortam yaratmasına yardımcı olan süreç odaklı, sağduyulu uygulamalardır.

ISO 9001, Kalite Yönetim Sistemleri—Gereksinimler, ürün gerçekleştirme tartışması içinde gereksinim yönetimi konusunda bazı rehberlik sağlar. Standart, aşağıdakilerin belirlenmesini gerektirir:

1. Ürün için kalite hedefleri ve gereksinimleri
2. Süreçler, belgeler oluşturma ve özel kaynaklar sağlama ihtiyacı ürün
3. Ürüne özel gerekli doğrulama, geçerli kılma, izleme, muayene ve test faaliyetleri ve ürün kabul kriterleri
4. Gerçekleştirme süreçlerinin ve ortaya çıkan ürünün gereksinimleri karşıladığını kanıtlamak için gereken kayıtlar (ISO 9000 2015)

ISO 9001, gereksinim yönetimi için özel süreç kılavuzu sağlamaz. Bununla birlikte, yukarıda belirtilen tavsiyeler, özellikle uygun gereksinim metrikleriyle birlikte kullanıldığında bir kontrol listesi olarak faydalıdır (bunlardan bazıları Bölüm 8'de tartışılmıştır). ISO standardı kapsamında sertifikasyon elde etmek için önemli belgeler gereklidir.

ISO/IEEE 12207

ISO 12207: Bilgi Teknolojisi Standardı—Yazılım Yaşam Döngüsü Süreçleri, beş temel süreci tanımlar: edinme, tedarik, geliştirme, bakım ve çalıştırma. ISO 12207, beş süreci faaliyetlere ve faaliyetleri görevlere ayırırken, bunların yürütülmesi üzerine gereksinimler koyar. Aynı zamanda sekiz destekleyici süreci (dokümantasyon, konfigürasyon yönetimi, kalite güvencesi, doğrulama, doğrulama, ortak inceleme, denetim ve problem çözme) ve ayrıca dört organizasyonel süreci (yönetim, altyapı, iyileştirme ve eğitim) belirtir.

ISO standardı, kuruluşların bu süreçleri, uygulanamayan tüm faaliyetleri silerek kendi özel projelerinin kapsamına uyacak şekilde uyarlamalarını amaçlar ve ISO 12207 uyumluluğunu bu süreçlerin, faaliyetlerin ve görevlerin uyarlanmış performansı olarak tanımlar.

ISO 12207, belirli bir yaşam döngüsü modelini veya yazılım geliştirme yöntemini dikte etmek yerine, karşılıklı olarak kabul edilen terminolojiyi kullanan bir süreç yapısı sağlar. Nispeten üst düzey bir belge olduğu için 12207, süreçleri oluşturan faaliyetlerin ve görevlerin nasıl gerçekleştirileceğine ilişkin ayrıntıları belirtmez. Belgelerin adını, biçimini veya içeriğini de belirtmez. Bu nedenle, 12207'yi uygulamak isteyen kuruluşların, bu ayrıntıları belirten ek standartlar veya prosedürler kullanması gerekir.

IEEE, bu standardı eşdeğer numaralandırma ile tanır: IEEE/EIA 12207.0-1996, IEEE/EIA Uluslararası Standart Endüstri Uygulaması

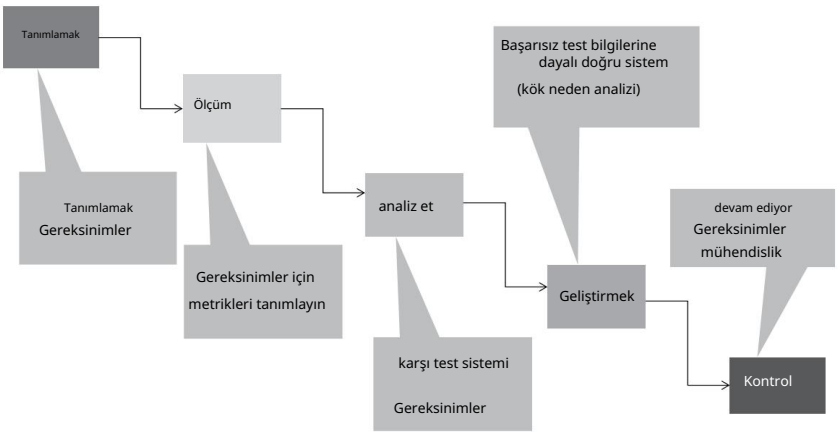
Standart ISO/IEC12207:1995 ve (ISO/IEC 12207) Bilgi Teknolojisi Standardı—Yazılım Yaşam Döngüsü Süreçleri. ISO/IEC 15504 Bilgi Teknolojisi—SPICE olarak da adlandırılan Süreç Değerlendirmesi, 12207'nin bir türevidir.

Altı Sigma

Motorola tarafından geliştirilen Altı Sigma, süreç varyasyonlarını ortadan kaldırmaya dayalı bir yönetim felsefesidir. Altı Sigma, çıktıların belirlenen hedeflerin ortalamasından altı standart sapma (altı sigma) içinde olmasını sağlamak için bir sürecin kontrolüne odaklanır. Altı Sigma, DMAIC tanımla, ölç, analiz et, iyileştir ve kontrol et. Gereksinim yönetimi için bir Altı Sigma modeline genel bakış Şekil 9.3'te gösterilmektedir.

Altı Sigma'da "tanımlamak", genellikle bir tür iş süreci modeli kullanarak iyileştirilecek süreci tanımlamak anlamına gelir. Gereksinim mühendisliği durumunda, bunu iyileştirmeye ihtiyaç duyan gerekli özellikleri belirlemek olarak düşünebiliriz. "ölçüm", süreç modelinin her yönü için, bizim durumumuzda tanımlanan gereksinimler için ilgili metrikleri belirlemek ve yakalamak anlamına gelir. Hedef soru-metrik paradigması bu konuda yardımcı olur. "Analiz", iyileştirme fırsatları için yakalanan metrikleri incelemek anlamına gelir. "İyileştirme", genellikle en yüksek geri ödemeye sahip olacak yöne saldırarak, ilişkili ölçümlerde faydalı değişikliklerin görülmesi için sürecin bazı yönlerini değiştirmek anlamına gelir. Son olarak, "kontrol", modeli sürekli olarak yeniden ziyaret etmek, metrikleri gözlemlemek ve süreci gerektiği gibi iyileştirmek için metriklerin sürekli izlenmesini kullanmak anlamına gelir.

Altı Sigma, CMMI'den daha fazla süreç verimine dayalıdır, bu nedenle CMMI süreç alanları, DMAIC'yi desteklemek için kullanılabilir (örneğin, ölçümü teşvik ederek).
Ve CMMI faaliyetleri tanımlarken, Altı Sigma bu faaliyetlerin optimize edilmesine yardımcı olur.



Şekil 9.3 Gereksinim Mühendisliği için Altı Sigma Süreci.

Altı Sigma, CMMI uygulamalarının uygulanması için özel araçlar da sağlayabilir (örneğin, tahmin ve risk yönetimi).

Bazı kuruluşlar, yazılım kalitesi uygulamalarının bir parçası olarak Altı Sigma kullanır. Ancak buradaki sorun, basit ve son derece yapay bir şelale sürecine dönüşmeyen yazılım üretim süreci için uygun bir iş süreci modeli bulmaktır. Gereksinimler için uygun metrikler belirlenebilirse (örneğin, Bölüm 8'de tartışılanlar), YE sürecini iyileştirmek için Altı Sigma kullanılabilir.

VIGNETTE 9.1 FBI Sanal Vaka Dosyası

ABD Federal Soruşturma Bürosu'nun (FBI) feci sanal dava dosyası (VCF) sistemi, diğer şeylerin yanı sıra, gereksinim yönetiminin olağanüstü bir başarısızlığıydı. Bu tartışma büyük ölçüde Goldstein'ın raporuna (2005), Alfonsi'nin analizine (2005) ve İsrail'in proje retrospektifine (2012) dayanmaktadır.

2000 yılında başlayan VCF'nin FBI'ın kağıt tabanlı çalışma ortamını otomatikleştirmesi ve ajanların ve istihbarat analistlerinin soruşturma bilgilerini paylaşmasına izin vermesi gerekiyordu. Sistem ayrıca 1970'lerin yazılım teknolojisine dayanan eski otomatik vaka desteğinin (ACS) yerini alacaktı. 12 yıl ve 451 milyon dolardan sonra proje tamamlanmadı ve asla tamamlanmayabilir. VCF sonunda tam olarak faaliyete geçebilir, ancak tüm hesaplara göre proje mali ve politik bir başarısızlık oldu.

VCF'de ne yanlış gitti? Oldukça, bazıları politik, bazıları devasa bir bürokrasinin karmaşıklığından kaynaklanıyor, ancak bazı başarısızlıklar kötü gereksinim mühendisliğine atfedilebilir. Goldstein, projenin başından beri yetersiz tanımlandığını ve yavaş yavaş gelişen gereksinimlerden (başlangıçta 800 sayfalık bir belge) muzdarip olduğunu kaydetti. Gereksinim keşfi için JAD oturumları kullanılmış olsa da (6 ay boyunca 2 haftalık birkaç oturum), bunlar projenin sonlarında ve yalnızca satıcı değişikliğinden sonra kullanıldı. Görünüşe göre, kartopu gerekliliklerini durduracak bir disiplin yoktu. Sürekli değişen bir dizi sponsor (FBI'daki liderlikteki değişiklikler ve ayrıca önemli kongre destekçilerindeki devir) gereksinimlerin kaymasına neden oldu.

Ayrıca, görünüşe göre, SRS'de, düğmelerin ve metnin rengini ve konumunu tam anlamıyla tanımlayan çok fazla tasarım detayı vardı. Ve birçok gereksinim IEEE 29148 kurallarında başarısız oldu—çoğu zaman "net, kesin ve eksiksiz" değillerdi. İsrail (2012) projenin kapsamlı prototiplemeyi içermesi gerektiğini, ancak çalışan işlevsellik sağlamaya yönelik baskıların prototip oluşturmayı imkansız hale getirdiğini belirtti.

Son olarak, FBI'da kültürel sorunlar da vardı - fikirleri paylaşmak ve zorlu kararlar norm değil. Bu faktörlerin tümü, herhangi bir ortamda, herhangi bir gereksinim mühendisliği çabasının etkinliğini tehlikeye atacaktır.

Projenin daha sonraki bir incelemesi, suçlanacak tek bir kişi veya grup olmadığını kaydetti - bu her düzeyde bir başarısızlıktı. Ancak bağımsız bir gerçek bulma komitesi, daha önce belirtilen nedenlerle FBI liderliğini ve ana yükleniciyi hatalı olarak seçti (Goldstein 2005).

Proje karmaşık bir projeydi ve eski FBI CIO'su İsrail'e göre organizasyonun yönetme yeteneklerinin ötesindeydi (İsrail 2012). Proje, CIO'ların ve proje liderlerinin birden fazla değişikliğinden geçti. Başlangıçta dahili bir proje olarak başlayan FBI, bunu tamamlamak için gerekli uzmanlığa sahip olmadığını fark etti ve 2007'de projeyi Lockheed Martin'e devretti. Ancak 2010 yılına kadar birçok aksilikten sonra FBI projeyi geri aldı ve bu yazıda projeyi çevik bir yaklaşımla tamamlamayı bekliyor (İsrail 2012).

Büyük karmaşık sistemler tasarlanırken ortaya çıkan siyasi sonuçlar ve olağan problemler dışında, VCF'nin başarısızlığından ne öğrenilebilir? Gereksinim mühendisliği açısından bakıldığında, çok özel üç ders vardır. İlk olarak, ortaya çıkarma tamamlanmadan bir sistemi çıkarmak için acele etmeyin.
Açıkçası, özellikle bu kadar pahalı ve yüksek profilli bir sistem için yapılması gereken baskılar var, ancak bu baskılara güçlü bir liderlik tarafından direnilmesi gerekiyor. İkincisi, sistemin baştan tam ve doğru bir şekilde tanımlanması gerekir.
Evrimi sırasında FBI liderliğindeki çeşitli değişiklikler nedeniyle bu VCF için oldukça sorunlu olsa da, iyi disiplinli bir süreç, geç gereksinim değişikliklerine karşı geri itme sağlamaya yardımcı olabilirdi. Son olarak, çok büyük sistemler için bir mimarinin akılda tutulması yararlıdır. Mimarinin gereksinim belirtimine açıkça dahil edilmemesi gerektiğinin söylendiğinin farkındayız, ancak mimari bir hedefe sahip olmak, hayal edilen mimari nihai olmasa bile, ortaya çıkarma sürecini bilgilendirmeye yardımcı olabilir.

Egzersizler

9.1 Özellik ekleme veya değiştirme isteği anonim mi olmalı? Neden veya neden olmasın?

9.2 Aşağıdaki spesifikasyon için, olası gereksinimleri tanımlayın.

2 yıllık bir süre içinde modası geçmiş hale gelir:

9.2.1 Ek A'daki akıllı ev sistemi

9.2.2 Ek B'deki ıslak kuyu kontrol sistemi

9.3 Bir kuruluşta metrik kötüye kullanımı nasıl gelişmeye başlayabilir?

9.4 Organizasyonlarda süreç sınıfının bazı kaynakları nelerdir?

9.5 Kendi deneyiminizden bir süreç çatışması örneği verin.

9.6 Kendi deneyiminizden yola çıkarak, metriklerin kötüye kullanılmasına bir örnek verin.

9.7 Kuruluşlardaki farklı hedeflerin bazı kaynakları nelerdir?

9.8 Kendi deneyiminizden farklı hedeflere bir örnek verin.

9.9 CMMI, süreç çakışmasını belirlemek ve uzlaştırmak için nasıl kullanılabilir?

9.10 Bir grup arkadaş, sınıf arkadaşı veya takım arkadaşıyla, aşağıdaki seçenekler arasından bir yemek paylaşmak üzere bir restoran seçmek için Geniş Bant Delphi'yi kullanın:

9.10.1 Zeytin Bahçesi

9.10.2 Carrabas

9.10.3 Kırmızı Istakoz

9.10.4 Altın Ağıl

9.10.5 PF Chang's

9.10.6 Ruth'un Chris Biftek Restoranı

9.11 Beş gereksinim yönetimi aracından oluşan bir set seçin. Web sitelerinde yayınlanan bilgilerden, gereksinim mühendisliğinin yönetim yönlerini destekleyen özelliklerin bir listesini oluşturun.

9.12 7'deki karar verme problemini çözmek için analitik hiyerarşi sürecini (AHP) araştırın ve kullanın.

Referanslar

Alfonsi, B. (2005). Arafta yaşayan FBI'ın sanal dava dosyası, Güvenlik ve Gizlilik, 3(2): 26-31.

Andriole, S. (1998). Gereksinim yönetimi politikası. IEEE Yazılımı, Cilt 15, 82–84.

Bhat, JM, Gupta, M. ve Murthy, SN (2006). Gereksinim mühendisliği zorluklarının üstesinden gelmek: Açık denizde dış kaynak kullanımından alınan dersler. IEEE Yazılımı, Cilt 23, 38–44.

Brown, WJ, Malveau, RC, McCormick, HW ve Mowbray, TJ (1998). AntiPatterns: Krizdeki Yazılımları, Mimarileri ve Projeleri Yeniden Düzenleme. Wiley. New York, NY.

Dekkers, CA ve McQuaid, PA (2002). Yazılım ölçümlerini kullanmanın tehlikeleri (yanlış) üstesinden gelmek. BT Uzmanı, 4(2): 24-30.

Ebert, C. (2010). Gereksinim mühendisliği: Yönetim. P. Laplante'de (Ed.), Yazılım Mühendisliği Ansiklopedisi, s. 932–948. Taylor ve Francis. Boca Raton, Florida.

Goldstein, H. (2005). Sanal dava dosyasını kim öldürdü? Spektrum, 42(9): 24-35.

Hull, E., Jackson, K. ve Dick, J. (2011). Gereksinim mühendisliğinin yönetim yönleri. Gereksinim Mühendisliği, 159–180. 2. baskı Springer-Verlag, Londra.

ISO 9000 Kalite Yönetim Sistemleri. (2015). http://www.iso.org (Ocak'ta erişildi 2017).

İsrail, JW (2012). FBI neden bir vaka yönetim sistemi kuramıyor? Bilgisayar, 45(6): 73-80.

Kasab, M. (2014, Haziran). AHP tarafından kalite gereksinimleri için erken çaba tahmini. Uluslararası Hesaplamalı Bilim ve Uygulamaları Konferansında, s. 106–118, Springer International Publishing.

Laplante, PA (2006). Her Mühendisin Yazılım Mühendisliği Hakkında Bilmesi Gerekenler. ÇHS/Taylor & Francis.
Lawrence, B. (1996). Çözülmemiş belirsizlik. Amerikalı Programcı, 9(5): 17–22.
Mahaux, M. ve Maiden, N. (2008). Tiyatro doğaçlamacılar oyunun gerekliliklerini bilirler. Yazılım, Cilt 25, 68–69.
Neill, CJ, Laplante, PA ve DeFranco, J. (2012). Antipatterns: Tanımlama, Yeniden Düzenleme, Ve yönetim. 2. baskı CRC Basın. Boca Raton, Florida.
Norden, B. (2007). Gereksinim mühendisleri için senaryo yazımı. Yazılım, Cilt 24, 26–27.
Paulk, M., Weber, CV, Garcia, SM, Chrissis, M.-BC, & Bush, M. (1993). CMM Uygulamaları Kılavuzu. CMU/SEI-93-TR-25. https://www.sei.cmu.edu/reports/93tr024. pdf (Şubat 2017'de erişildi).
Reinhart, G. ve Meis, JF (2012). Eşzamanlı mühendislik için bir başarı faktörü olarak gereksinim yönetimi. Üretimde Rekabetçiliğin ve Ekonomik Sürdürülebilirliğin Sağlanması, 221–226. Springer, Berlin.
Sinha, V. ve Sengupta, B. (2006). Dağıtılmış gereksinim yönetiminde işbirliğini etkinleştirme. Yazılım, Cilt 23, 52–61.