Java Server Faces Tabanlı Proje Akış Yönetim Sisteminin Geliştirilmesi: ODTÜ PAYS

Ayberk KOÇ, Çağıl ŞEKER
Orta Doğu Teknik Üniversitesi, Ankara
ayberk@metu.edu.tr, cagil@metu.edu.tr

Özet: Bu çalışmada, ODTÜ bünyesinde geliştirilen Proje Akış Yönetim Sistemi hakkında bilgi verilecektir. Bu kapsamda, Java uygulama geliştirme dili ve JSF çatısı ile geliştirilen bu sistemin genel yapısıyla beraber, kullanılan uygulama geliştirme ortamına, uygulama mimarisine ve içeriğine yer verilmektedir.

Giriş

2005 yılı başı itibarı ile kullanılmakta olan sistem, gereksinimlerin zaman içinde değişmesi, tüm kullanıcı profillerinin kapsanmaması ve sistemin iç yapısının yeniden düzenlenerek ihtiyaca göre güncellenmesinin zorluğu (refactoring) nedeni ile işlevselliğini yavaş yavaş yitirmeye başlamıştır. Bu nedenden ötürü proje/para giriş ve çıkış işlemleri (fiş kesimi ve muhasebeleştirme dahil) ile beraber, para akışının tamamen kontrol edilip raporlanabildiği yeni bir sisteme ihtiyaç duyulmuştur.

Bu doğrultuda 2005 yazından itibaren projeyi kullanacak birimlerden genel gereksinimler toplanmaya ve eş zamanlı olarak uygulama geliştirme ortamı ile mimarisinin nasıl olması gerektiği üzerinde de çalışılmaya başlanmıştır.

Yapılan incelemeler sonucunda geliştirme dili olarak Java, geliştirme ortamı olarak Eclipse üzerine yapılandırılmış olan IBM Rational yazılım geliştirme platformu (RSA) ve uygulama sunucusu olarak da Websphere Application Server v6.0'da karar kılınmıştır. Kullanılma olasılığı olan çatılar incelenmiş ve proje kapsamı, geliştirme süresi ve programcı deneyimi de göz önüne alınarak sistemin Java Server Faces (JSF) üzerine kurulmasının daha uygun olacağı öngörülmüştür.

Sistemin inşası için kullanılacak kütüphaneler :

- Uygulama temel çatısı olarak: JSF (IBM JSF uyarlaması)
- Kullanıcı doğrulama bağlantısının sağlanabilmesi için: LDAP (Novell LDAP)
- Raporlama için: PDF (JasperReports)

şeklinde belirlenmiştir.

Sistemin esnekliğini ve basitliğini korumak ve SQL sorgu dilinin tüm gücünü kullanabilmek adına Hibernate veya benzeri bir veri modeli katmanı kullanmak yerine basit bir bağlantı kapsayıcısı yazılarak doğrudan bağlantı havuzu üzerinden veri tabanı sorgulama işlemlerinin yapılması tercih edilmiştir.

Bunların yanı sıra uygulamadan alınabilen çeşitli raporlar ve formlar için kullanılan JasperReports açık kaynak raporlama aracı ile tüm dokümanlar PDF dosya formatında istenirse basıma hazır olarak elde edilebilmektedir.

Java Server Faces Çatısı

JSF'nin uygulamanın davranışı ile sunumu arasında net bir ayrım sağlaması web tabanlı uygulamaların geliştirilmesi için çok büyük bir avantaj sağlamıştır. Sadece Java Server Pages (JSP) teknolojisi kullanılarak bir dereceye kadar bu ayrımı sağlamak mümkün olsa da, JSP'nin kullanıcı arayüzü nesnelerine özel HTTP çağrılarını yanıtlayamaması ya da du-

rum (state) bilgisini kolayca koruyup yönetememesi gibi eksiklikleri nedeniyle JSP üstüne kurulu daha özelleşmiş bir çatıya gereksinim duyulmuştur.

JSF sunucu-taraflı bir kullanıcı arayüzü olarak temelde aşağıdaki özellikleri sağlamaktadır:

- Kullanıcı arayüzü nesnelerinin sayfa üzerine basit tag'lar ile kolayca yerleştirimi
- Bu nesneler tarafından üretilen olayların sunucu tarafında uygulama koduyla otomatik eşleştirimi
- Kullanıcı arayüzü içeriğinin sunucu-taraflı veriye bağlanması
- Nesnelerin durumunun sunucu istemlerinden daha uzun ömürlü olarak tutulabilmesinin sağlanması
- Kullanıcı girişlerinin geçerlenmesi konusunda geniş destek

JSF, doğrudan Servlet API'si üzerine inşa edilmiş bir çatı olarak JSP dışında başka herhangi bir sunum aracına da destek vermektedir. Hazır JSP Tag kütüphanesinin bulunması nedeniyle bu projede JSP arayüzü tercih edilmiştir.

Genel hatlarıyla bakıldığında bir JSF uygulaması aşağıdaki öğelerden oluşmaktadır:

- JSP sayfaları
- 'Backing Beans'
- Uygulama konfigürasyon kaynak dosyası (faces-config.xml)
- 'Deployment Descriptor' (web.xml)
- Özel (Custom) nesneler (Bileşenler (Components), Geçerleyiciler (Validators), Dönüştürücüler (Converters), ve Dinleyiciler (Listeners))
- Özel (Custom) etiketler (Tags) (Bileşenleri sayfa üzerinde gösterebilmek için)

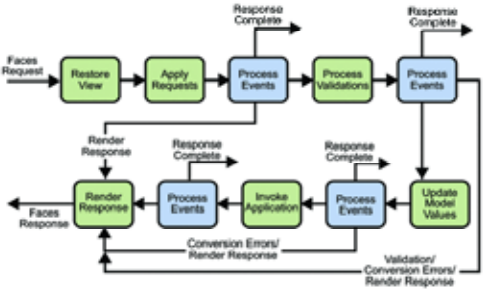
JSF'nin mimarisi bir web uygulamasının net tanımlanmış roller tarafından parça parça geliştirilmesine de olanak tanımaktadır. Bu roller örneğin şu şekilde olabilir:

- Uygulama Mimarı: Konfigürasyon, navigasyon ve entegrasyonla uğraşır.
- Bileşen Geliştiricisi: Gereken özel bileşenleri geliştirir.
- Uygulama Geliştiricisi: Özel nesneleri ve 'Backing Bean'leri geliştirir.

JSP sayfaları ile Backing Bean'ler arasındaki bağlantı Expression Language (EL) ile yapılmaktadır. JSF özel olarak bu dilin 'Deferred' (geciktirilmiş) sözdizimini kullanmaktadır.

Backing Bean mekanizmasına bağlı olarak çalışan Data Value Object (DVO) nesneleri ve bu nesnelerin tablolara bağlanması için DVO liste nesneleri oluşturularak Bean'lerin veri modeline bağlanması sağlanmıştır.

JSF yaşam döngüsüne bakıldığında, gelişmiş arayüz desteği nedeniyle basit bir JSP döngüsünden daha çok aşama içerdiği görülmektedir:



Şekil-1: JSF Yaşam Döngüsü

JSF yaşam döngüsü esas olarak altı ana adımdan oluşmaktadır. Bir JSF sayfası ilk yüklendiğinde sadece *Restore View* **ve** *Render Response* aşamaları çalıştırılmaktadır. Sayfa üzerindeki işlemler sonucu sayfa yeniden çalıştırıldığında ise tüm aşamalar gerçekleştirilmektedir.

JSF'de kullanıcı girdilerinin gerçerlemesi (validation) ayrı bir adımdır ve önceden tanımlanan geçerleme nesneleri ve kriterleri tarafından kontrol edilir. Geçerlemeden geçemeyen

girdiler için hata mesajları otomatik olarak yaratılarak sayfa üzerinde nesneyle ilişkili olarak istenildiği gibi yerleştirilebilmektedir. JSF aynı zamanda hata veya başka bir nedenle sayfanın yenilenmesi durumunda form üzerindeki kullanıcı girdilerinin otomatik olarak eski durumlarını korumalarını sağlayarak geliştirmeye kolaylık kazandırmaktadır.

Geliştirme Ortamı

IBM Rational Software Architect hızlı uygulama geliştirmeye olanak sağlaması ve uygulama geliştirme sürecinin tüm aşamalarını kapsaması dolayısı ile için tercih edilmiştir. Bunun ile beraber kaynak kontrolü ve etkinlik yönetimini sağlayabilmek amacı ile Rational ClearCase ve Rational ClearQuest araçları kullanılmıştır.

Uygulama

Mimari ve istenilen çatı ayarlandıktan sonra kullanıcı doğrulama işlemleri için ilgili altyapı tasarlanmış ve sistemin ODTÜ kullanıcı verilerine doğrudan ulaşması sağlanmıştır. Bu doğrultuda kurulan yapıda kullanıcıların profilleri ve yapabilecekleri işler de belirlenerek ilgili doğrulama ve yetkilendirme işi tamamlanmıştır (Şekil-3).

Şekil-3. Kullanıcı Yetki Seçme Sayfası

Uygulama şu an için bilimsel çalışma tanımlama, bilimsel çalışmadan harcama yapma (avans, satınalma, yolluk, ödeme çizelgesi v.b.), para giriş ve çıkışlarının tanımlanması, fiş kesimi ve muhasebeleştirilmesi, işlem onaylarının yapımı ve süreçlerinin takibi ile beraber kullanıcıların resmi evrak formatında kullanmakta olduğu formlar ve öngördüğü raporları da pdf olarak kullanıcılara sunabilmektedir.

Ek olarak profillerce belirlenmiş işlerin yanında, kullanıcılar belirli iş yükleri nedeniyle yaptıkları gündelik işleri başka kullanıcılara da atayabilirler. Uygulama A kişisinin B kişisine yetki vererek belirli bir iş başlığını onun adına yapabilmesine olanak sağlamaktadır. Örneğin bir proje yöneticisi bu iş için görevlendirilmiş bir kişiye kendi projelerinden harcama yapma yetkisi verebilir. İlgili kişi de kendi profilinin yanında kendisine tanımlanmış yetkileri görebilir ve profilini atanmış yetkiye değiştirerek ilgili işi A kişisi adına yapabilir (Şekil-3). Bu süreçte güvenlik sebebi ile yapılmış iş ve A kullanıcısı ile beraber kişinin esas profilinin de işle ilgili kaydı **(***log***)** tutulur.

Sonuç

ODTÜ PAYS projesinde Java Server Faces çatısının birleştirici ve esnek modeli, geliştirme ortamının otomasyon desteği ile birleştirilerek, geliştirme sürecinin tüm aşamaları boyunca harcanan emeğin azaltılması, kontrol akışının kolayca yönetilmesi ve ortak bir kullanıcı arayüzü modeli sunulabilmesi sağlanmıştır.