10. Pompa Vana Kompresör Kongresi
13-14 Şubat 2020, Ankara

TERMAL HİDROLİK SİSTEMLERİN DİJİTAL İKİZLERİNİN OLUŞTURULMASI İÇİN NESNE YÖNELİMLİ BİR YAZILIM GELİŞTİRİLMESİ

Onur Tunçer
Prof.Dr.
İstanbul Teknik Üniversitesi
Uçak Mühendisliği Bölümü
tuncero@itu.edu.tr

Stanislav Grate
Uzay Mühendisi
Melina Aero Teknoloji Geliştirme ve Dizayn Bürosu A.Ş.
stanislav.grate@melina-aero.com

Yekta Mirkan
Elektrik-Elektronik Mühendisi
Melina Aero Teknoloji Geliştirme ve Dizayn Bürosu A.Ş.
yekta.mirkan@melina-aero.com

Ferit Tunçer
Bilgisayar Mühendisi
Melina Aero Teknoloji Geliştirme ve Dizayn Bürosu A.Ş.
ferit.tuncer@melina-aero.com

Orhan Topdağ
Bilgisayar Mühendisi
Melina Aero Teknoloji Geliştirme ve Dizayn Bürosu A.Ş.
orhan.topdag@melina-aero.com

ÖZET

Termal hidrolik sistemlerin bilgisayar ortamında dijital ikizlerinin oluşturulması için FlowNetMaster ismi ile yeni bir yazılım geliştirilmiştir. FlowNetMaster tamamen nesne yönelimli programlama paradigmaları ile C# dilinde .NET Framework teknolojisi ile model-gösterim-gösterim modeli (MVVM) mimari yapısı kullanılarak geliştirilmiştir. İçerdiği veritabanı sayesinde standart komponentler (borular, pompalar, fitting elemanları) kullanıcı tarafından kolayca seçilebilmekte ve istenildiğinde kullanıcı tarafından yeni komponentler tanımlanabilmektedir. Modelin oluşturulması için kullanıcı kütüphanelerden seçtiği komponentleri (örneğin ısı değiştiricisi, boru v.b.) sürükle bırak etkileşimlerle çizim tuvaline yerleştirmekte daha sonra ise bu ortamda komponentlerin birbirlerine bağlantılarını yapmaktadır. Yazılım bu modeli çekirdek katmanında (model katmanı) bir diferansiyel cebirsel denklem takımına dönüştürmekte ve modelin sürekli ve geçici rejimlerdeki davranışını çözebilmektedir. Yazılımın standart termo-fiziksel kütüphanesinde yüzden fazla akışkan (su, hava, soğutma gazları ve sıvıları, nemli hava, antifriz çözeltileri v.s.) tanımlıdır. Ayrıca kullanıcıya da kendi istediği akışkanı tanımlayabilme özelliği sunulnuştur. Yazılımın sahip olduğu gösterge paneli (insan makine arayüzü) özelliği sayesinde kullanıcılar model ile etkileşebilmekte ve model çalışırken göstergeleri takip edebilmektedirler.

Geliştirilen yazılım görsel arayüzden kullanılabildiği gibi bir API (uygulama programlama arayüzü) üzerinden Python veya C# üzerinden de modeler oluşturulup çözülebilmektedir. API için son kullanıcı dokümantasyonu otomatik oluşturulmakta ve böylece sürekli güncel tutulmaktadır.

Yazılım modüler bir yapıda geliştirilmiş olup içerisinde akışkan, sinyal, mantık, ısı geçişi, mekanik, güç, tasarım optimizasyonu, uygulama programlama arayüzü v.b. kütüphanelerden oluşmaktadır. Bu modüllerin ayrı ayrı veya birlikte paketlenmesi mümkündür.

Yazılım geliştirme sürecinin yönetimi için bulut üzerinde sürekli iyileştirme ve sürekli dağıtım stratejileri kullanılmaktadır. Böylece son kullanıcı isteklerini anında tatmin edebilecek bir entegrasyon kurulmuştur. Bu entegrasyon otomatik testleri de içermektedir. Böylece yazılımın bütünlüğü ve sonuçların doğruluğu yazılım kullanıcıya ulaşmadan sürekli test edilmektedir.

Bildiride yazılım geliştirme süreci ile mimari yapı kısaca ele alınmakta ve çeşitli kütüphaneler için seçilen muhtelif doğrulama örnekleri sunulmaktadır. Programın çıktılarının literatürdeki verilerle uyumlu olduğu görülmektedir. Dolayısıyla programın endüstriyel vana, kompresör, pompa sistemlerini ve bunlara ait denetleyicileri modellemekte ve optimize etmekte kullanılabileceği sonucuna varılmıştır.

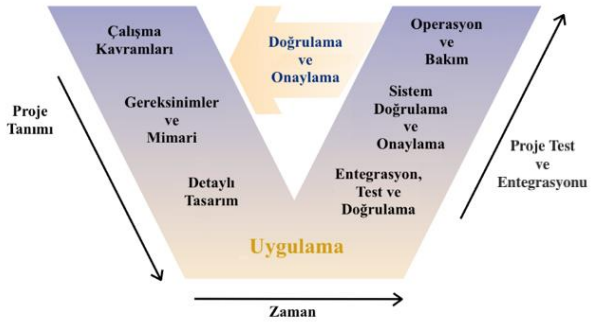
SEMBOLLER

A : Katsayı
A : Alan (m^2)
B : Katsayı

c : Isı sığası (J/kg.K)
C : Kapasitans (Farad)
C_i : Katsayı
F : Frekans (Hz)
f : Fonksiyon
g : Fonksiyon
i : İndis
k : Ayrıklaştırılmış zaman adımı
k : Isı iletim katsayısı (W/m.K)
L : İndüktans (Henry)
n : Eğri uydurma için nokta adedi
P : Basınç (Pa)
R : Direnç (Ohm)
R : Kirlenme faktörü, (m^2.K/W)
ref : Referans
s : Entropi (J/K)
t : Zaman (saniye)
T : Sıcaklık (K)
u : İç enerji (J/kg)
V : Voltaj (Volt)
y : Durum değişkeni vektörü
Δt : Zaman adımı (saniye)
μ : Viskozite (Pa.s)
τ : Zaman sabiti (saniye)

GİRİŞ

Mühendislik sistemlerinin tasarımı ve geliştirilmesi, ki termal-hidrolik sistemler de birer mühendislik sistemidir, çok farklı disiplinlerden uzmanların ortak çalışmasını gerektirir. Eş-zamanlı şekilde sistemin ön tasarım aşamalarında gerek disiplinler arası bilgi ve gereksinim paylaşımı gereklidir. Ancak bu sayede oluşturulan ön tasarım evrilerek başarılı bir ürüne dönüşebilir. Disiplinler arası mühendislik çalışmaları için sistem temelinde model tabanlı tasarım yaklaşımı, taslak oluşturmayı ve farklı ekiplerin ortak çalışmasını kolaylaştıran bir modeldir. Böylece ihtiyaçlar üst seviyede belirlendikten sonra model tabanlı tasarım ile disiplinler arası ekipler (akış, ısı, mekanik, elektrik-elektronik) sistemin farklı parçaları üzerinde aynı anda beraber veya birbirlerinden bağımsız olarak Şekil 1'de gösterilen V-döngüsü süreci ile çalışabilmektedir.

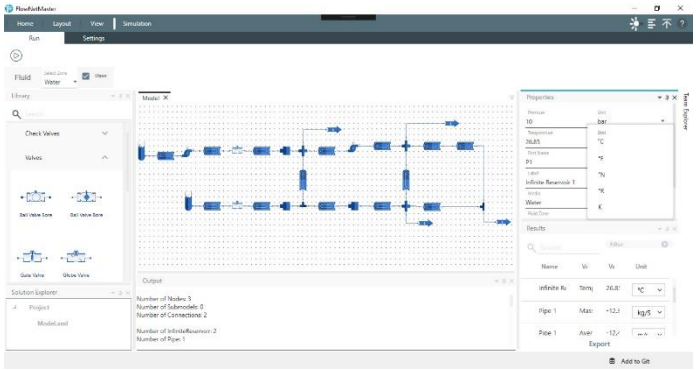


Şekil 1. Model Tabanlı Tasarım V-Döngüsü Geliştirme Süreci (1)

Bahse konu olan sistemler karmaşık olduklarından hesaplamaların elde yapılması mümkün değildir. Hesaplamalar ve optimizasyon için muhakkak suretle bir yazılım aracının kullanılması gerekmektedir. Bu makalede böyle bir yazılım aracının baştan aşağı geliştirme süreci ve doğrulama sonuçları ele alınmaktadır. Geliştirilen yazılıma ait bir ekran görüntüsü ise Şekil 2'de gösterilmektedir.

MODEL TABANLI TASARIM FELSEFESİ

Model tabanlı tasarımın amacı, tasarımcıların etkileşimli yazılım uygulamalarını uygulama düzeyini derhal başlatmak yerine daha semantik odaklı biçimde analiz etmesine olanak tanıyan üst seviye modelleri tanımlamaktır (3). Bu esasında karmaşık kontrol, sinyal işleme ve haberleşme sistemlerinin tasarımı ile ilgili problemlerin odaklanması için matematiksel ve görsel bir yöntem olarak tanımlanmaktadır (4-6). Bu yöntemle, tasarımcılar sürecin daha önemli yönlerine konsantre olurlar. Dolayısıyla, geliştirme ve iyilieştirme süreçleri boyunca sık yapılan analizler sayesinde süreç daha sorunsuz bir şekilde yönetilebilmektedir. Buna ek olarak, tasarımcılar kısıtları karşılamak ve en iyi dengeyi bulmak için de bu yöntemi kullanabilirler (7).



Şekil 2. Geliştirilen FlowNetMaster Programının Grafik Arayüzünden Bir Ekran Görüntüsü

Model tabanlı tasarım, diğer geleneksel yaklaşımlara kıyasla önemli avantajlar sunmaktadır (3,5). Ayrıca bu yaklaşım genel iletişim, veri analizi ve geliştirme ekipleri arasındaki sistem doğrulamalarını kolaylaştıran ortak tasarım ortamı sunmaktadır (8-10). Dahası mühendislerin, sistemsel değişikliklerin zamansal ve maddi etkileri azaltıldığında sistem tasarımının erken evrelerindeki hataları bulmaları ve düzeltmeleri mümkündür (8-10). Bunun yanı sıra, teknik iyileştirme ve türev sistemlerin kapasitelerini genişletmek için tasarımın yeniden kullanılması da kolaylaştırılmış olur (10). Bu teknikle, ürünler piyasaya daha hızlı ulaşır ve geleneksel yöntemlere kıyasla daha düşük maliyetli olur (13). Özellikle havacılık, otomotiv, ısıtma-soğutma uygulamaları arasında bu yöntemin kullanımı yaygındır, aynı zamanda gömülü yazılım tasarımı için de sıklıkla uygulanan bir metodolojidir (9,12,13).

Model tabanlı tasarım, bir yandan geliştirme sürecini desteklerken, bir yandan da tasarım süreci boyunca değişik

disiplerden takımlar arasında iletişim için ortak bir yapı oluşturur (8). Ürünün kontrol sistemlerinin entegrasyonu da model tabanlı tasarım ile yapılabilir. Bu, ürünün modellenmesi sonra denetleyici analizi ve sentezini, çevrimdışı ile gerçek zamanlı simülasyonları, nihayetinde de gömülü kodun oluşturulup dağıtımını da içeren tümleşik bir yaklaşımdır (9,13,14).

Geleneksel tasarım yöntemleri ile model tabanlı tasarım paradigması arasındaki bir diğer fark ise karmaşık yapılar ve kapsamlı yazılım kodu yerine, model tabanlı tasarım yaklaşımının sürekli ve ayrık zamanlı yapı blokları kullanarak gelişmiş ve işlevsel özelliklere sahip modelleri tanımlayarak kullanıyor olmasıdır. Benzetim yazılımları ile oluşturulmuş modeller, hızlı prototipleme, yazılım testleri ve sonuçları doğrulamaya yardımcı olabilir. Test ve doğrulama süreci gelişmiş olmakla kalmaz, aynı zamanda bazı durumlarda, yeni tasarım paradigması ile sistemde arzu edilen dinamik etkilerin geleneksel tasarım yöntemine kıyasla daha hızlı ve verimli bir şekilde elde edilmesi de sağlanır.

YAZILIM GELİŞTİRME METODOLOJİSİ

Yazılım geliştirme süreci boyunca yazılım mimarisinin planlanması ve dökümantasyonu UML (birleşik modelleme dili) ile yapılmıştır. Metodoloji olarak ise kademeli geliştirme modeli uygulanmıştır. Bu yaklaşımda yazılım, kademeli olarak dizayn, uygulama ve test aşamalarından bir döngü şeklinde geçmektedir. Bu sayede çalışabilir bir yazılım erken elde edilmiş olup, ayrıca kullanıcı tarafından muhtemel karşılaşılan problemlere karşı tasarım değiştirmek daha kolay olmuşve küçük parçalar halinde uygulandığı için testler daha kolayca icra edilmiştir. Tüm bu yaklaşımlar sayesinde yazılım geliştirme aşamasındaki riskler ve yazılımın daha sonraki ömür çevrimi boyunca karşılaşılacak muhtemel riskler asgariye indirilmiştir.

Yazılımın her bir parçası "birim test" yöntemi ile her bir derleme işlemi sonrasında otomatik olarak test edilmektedir. Testlerin yazılımı ne oranda test edip kapsadığı sayılabilir şekilde ölçülmekte, böylece başarılı bir test sonucunun ne kadar güvenilir olduğu kestirilebilmektedir. Ayrıca fiziksel sistemlerin yazılım üstünde modeleri oluşturularak, elde edilen çıktıların seçilen fiziksel doğrulama modellerinin sonuçları ile karşılaştırılarak yazılımın modelleme ve simülasyondaki isabetliliği de ayrıca ölçülmüştür. Sonuçların bir kısmı bu bildiride okuyucu ile paylaşılmaktadır.

Geliştirme sırasında dağıtık versiyon denetleyici Git ve otomatik kaynak kodu inceleyicisi kullanılmıştır. Otomatikleştirilen statik kod inceleme sayesinde ekip içerisindeki değişik yazılımcıların aynı standartta kodlama yapmaları sağlanmakla kalmamış aynı zamanda kodun ömür çevrimi sürecince idame ettirilebilirliği ve okunabilirliği konusunda mümkün olan en yüksek standart sağlanmaya gayret edilmiştir. Yazılımın sürüm sayıları ise değişiklik mesajlarına bağlanıp semantik sürüm oluşturma kurallarına göre otomatik olarak atanmaktadır.

Rutin işlemler sürekli entegrasyon yöntemiyle entegre edilmiş ve Şekil 3'te gösterilen "GitHub Flow" iş akışı modeline uygun olarak konfigüre edilmiştir. Öyle ki, yazılımdaki en küçük değişiklik ardından tamamen otomatik olarak gerçekleşmekte işlemler sırasıyla şunlardır: Derlenme, testler, testlerin kapsayacılığının ölçülmesi, kaynak kodunun incelenmesi, önceki adımlarda başarı sağlandığı takdirde değişikliğin ana branşa dahil edilmesi, yayınlanan son başarılı değişikliğin sürüm sayısının hesaplanması, yayınlanan son başarılı değişiklik ile birlikte yazılımın paketlenmesi. Bu yaklaşıma sürekli entegrasyon, sürekli iyileştirme (CI/CD) adı verilmektedir. Bahsi geçen işler Github kod havuzu ile Microsoft Azure DevOps portalı entegre edilmek suretiyle otomatikleştirilmiştir.

Şekil 3. Kullanılan Yazılım Geliştirme İş Akışı

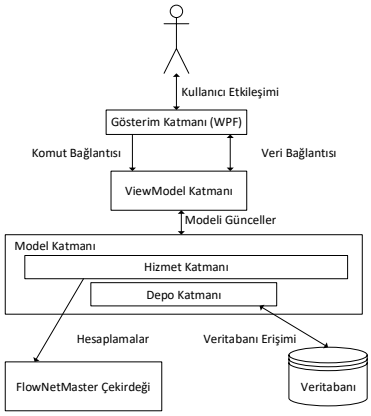
Ayrıca proje süresince düzenli olarak ayda bir takım içerisinde akran incelemesi yapılarak yazılımın kaynak kodu gözden geçirilip yeniden düzenlenmiş, böylece kaynak kodunun karmaşıklığı ile sürekli mücadele edilmek suretiyle ve verimliliğin düşmesi engellenmiştir.

KATMANLI YAZILIM MİMARİSİ

Her yazılım sisteminin tasarımı mimari kombinasyonlardan müteşekkildir. Yazılım temelde MVVM (model-view-view model) katmanlı mimarisi kullanılarak geliştirilmiştir. Katmanlı yazılım mimarisinin avantajı karmaşıklaşan yazılım isterlerinin standart bir yapıya uydurularak, bahsi geçen karmaşık yapıyı sorumluluklara, işlevlere ve görevlere göre birbirinden ayırmak suretiyle daha basit ve yönetilebilir hale gelmesini sağlamaktır. MVVM katmanlı mimari yaklaşımı yazılımın okunabilirliğini, sürdürülebilirliğini, ölçeklendirilebilirliğini ve test edilebilirliğini iyileştirmeyi amaçlar. Geliştirme sürecinde tercih edilen MVVM mimarisinin katmanları aşağıda sıralanmaktadır ayrıca Şekil 4'te yazılım mimarisi grafiksel olarak ta ifade edilmektedir.

- Model: Yazılımdaki veri ile iş mantığını temsil etmektedir. Hesaplamalar bu katmanda yapılmaktadır. Daha ileride bahsi geçecek uygulama programlama arayüzü (API) de bu katmanın erişimi sınırlandırılmış bir alt kümesine Python ve C# dilleri üzerinden erişebilmektedir. Ayrıntılar için (15) numaralı referansa müracaat edilebilir.
- View: Verinin kullanıcıya gösterilmesinden ve kullanıcıdan komut almakla sorumlu katmandır.

- ViewModel: Model ve View arasındaki bağlantıyı sağlar ve ilgili View'ı kontrol eder. Her bir View için bir ViewModel bulunmaktadır.



Şekil 4. FlowNetMaster Yazılım Mimarisi

Görsel arayüz için WPF (Windows Presentation Foundation) teknolojisi kullanılmıştır. Ayrıca yazılımda veri tabanına erişimi sağlamakla sorumlu bir "veri erişimi katmanı" bulunmaktadır. Veri tabanında Microsoft SQL Express kullanılmıştır. Veri tabanı sayesinde standart komponentlere kullanıcının hızlı erişiminin sağlanması amaçlanmıştır.

KÜTÜPHANE YAKLAŞIMI

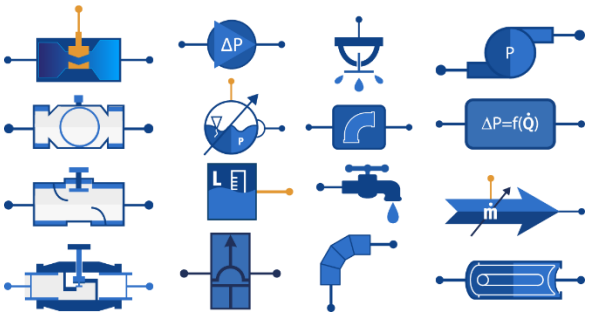
Yazılım mümkün olduğunca modüler bir yapı ile geliştirilmiştir. Benzer nitelikleri bağlantı arayüzlerine (ki yazılımın içinde bunlara "port" adı verilmektedir) sahip nesneler belirli kütüphanelere gruplanmışlardır. Bunlar sırasıyla;

- Akışkan
- Mantık
- Sinyal ve kontrol
- Isı transferi
- Elektrik
- Mekanik
- Güç

kütüphaneleri olarak sıralanabilir.

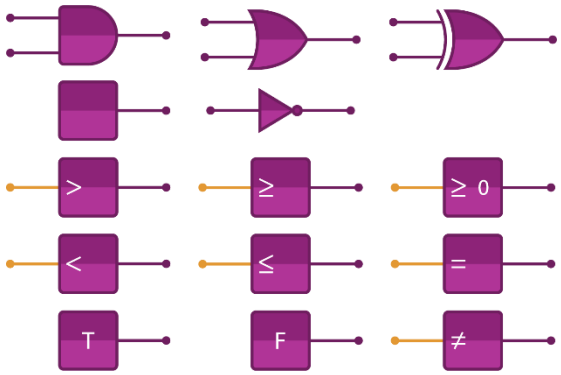
"View" katmanındaki her bir ikon model katmanında bir "sınıf" ile temsil edilmektedir, her sınıf kendi sürekli rejimdeki ve geçici rejimdeki davranışını tarif eden denklemleri modelin toplam denklem setine yazdırır.

Akışkan kütüphanesinde, pompalar, vanalar, borular, fitting elemanları v.b. komponentler yer almaktadır (Şekil 5).



Şekil 5. Akışkan Kütüphanesindeki Bloklardan Bir Kısmı

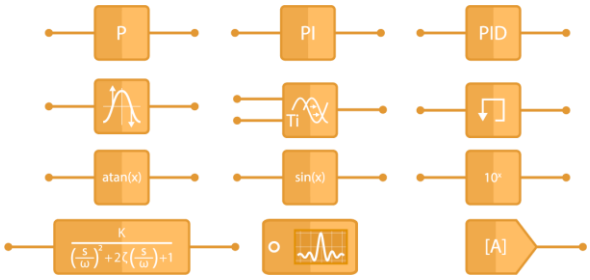
Mantık kütüphanesinde ise mantıksal operatörler ile karşılaştırma operatörleri bulunur (Şekil 6).



Şekil 6. Mantık Kütüphanesindeki Bloklardan Bir Kısmı

Sinyal kütüphanesinde sinyal kaynakları, fonskyonlar, basit filtreler, temel denetleyiciler (oransal, oransal-türev, oransal-türev-integral), osiloskop ve spektrum analizörü gibi görselleştirme araçları yer almaktadır (Şekil 7).

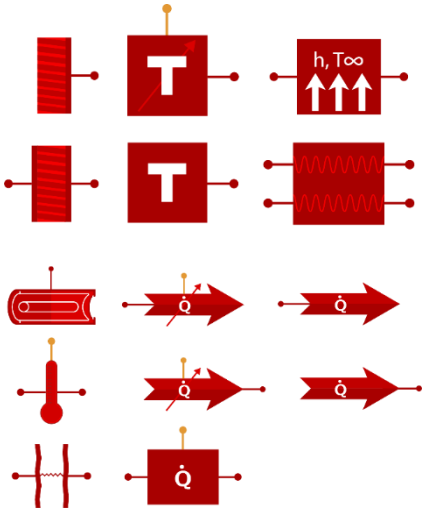
Isı geçişi kütüphanesindeki bloklardan bir kısmı Şekil 8'de gösterilmektedir.



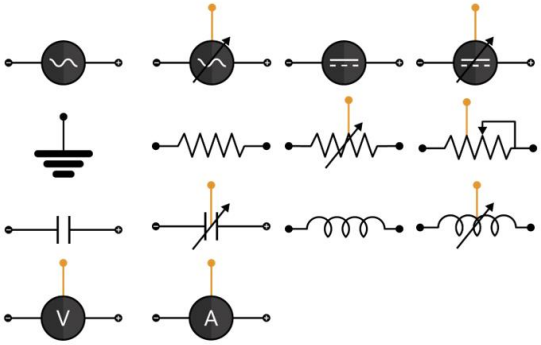
Şekil 7. Sinyal Kütüphanesi Bloklarından Bir Kısmı

Elektrik kütüphanesinde Şekil 9'da gösterildiği üzere pasif ve aktif analog devre elemanları bulunmaktadır.

Mekanik kütüphanesi hali hazırda şaft, dönel eylemsizlik, tork kaynağı gibi dönel mekanik elemanlardan müteşekkildir (Şekil 10).



Şekil 8. Isı Transferi Kütüphanesindeki Bloklardan Bir Kısmı



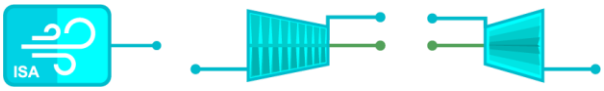
Şekil 9. Elektrik Kütüphanesindeki Bloklardan Bir Kısmı

Güç kütüphanesinde şimdilik atmosfer, kompresör, türbin gibi bloklar yer almaktadır (Şekil 11).

AKIŞKAN KÜTÜPHANESİ

Geliştirilen program geniş bir akışkan kütüphanesi barındırmaktadır. Bu kütüphanede yer almakta olan birçok akışkan CoolProp kütüphanesinin (16) yazılıma bağlanması sonucu kullanılmaktadır. Aşağıda hali hazırda yazılım tarafından desteklenilen akışkanların bir listesi bulunmaktadır.

Şekil 10. Mekanik Kütüphanesindeki Bloklardan Bir Kısmı



Şekil 11. Güç Kütüphanesindeki Blokların Bir Kısmı

- Su
- Oksijen
- Hidrojen
- Hava / Nemli Hava
- Amonyak
- Argon
- Benzen
- Dekan
- Metan
- Oktan
- İzo-Pentan
- Hidrojen-Sülfid
- Nonan
- Toluen
- Propan
- Nitrojen
- Pksilen
- Oksilen
- SF_6
- Karbondioksit
- Propilen
- Sikloheks
- Siklopen
- DME
- Etan
- Etanol
- Ebenzen
- Ağır Su
- Helyum
- Bütan
- H_2S
- İzo-Bütan
- Ipentan
- Mksilen
- Heptan
- Heksan
- Metanol
- Pentan

Soğutma sıvıları,

- R-134A
- R-116
- R218
- R-22
- R-227EA
- R-236FA
- R-12

- R-123
- R-1233ZD
- R-1234YF
- R-1234ZE
- R-245FA
- R-32
- R-404A
- R-407C
- R-410A
- R-507A
- R-124
- RC-318
- R-13
- R-14
- R-143A
- R-152A
- R-23
- R-236EA

Kütüphanedeki ısı transferi proses akışkanları ise şunlardır;

- Dietil Benzen
- Therminol D12 Solüsyonu
- Hydrofloroeter HFE-7100
- Polidimetilsiloksan 1
- Polidimetilsiloksan 2
- Sentetik Alkil Benzen
- Dynalene MV
- Terpen (Turunç Yağından)
- Aspen Temper -10
- Aspen Temper -20
- Aspen Temper -40
- Aspen Temper -55
- Zitrec S -10
- Zitrec S -25
- Zitrec S -40
- Zitrec S -45
- Zitrec S -55
- Therminol D12
- Therminol VP-1
- Therminol 72
- Therminol 66
- Dowtherm J
- Dowtherm Q
- Texatherm 22
- Nitrat Tuz Karışımı
- Syltherm XLT
- Dynalene HC-10
- Dynalene HC-20
- Dynalene HC-30
- Dynalene HC-40
- Dynalene HC-50

Kütüphanede yer alan sulu karışımların listesi ise aşağıda belirtilmektedir.

- Su - Etilen Glikol
- Su – Propilen Glikol
- Su - Etil Alkol
- Su - Metil Alkol
- Su - Gliserol
- Su - Amonyak
- Su - Potasyum Karbonat
- Su - Kalsiyum Klorid
- Su - Magnezyum Klorid
- Su - Sodyum Klorid
- Su – Potasyum Asetat
- Su - Potasyum
- Su - Lityum Klorid

Kütüphanedeki akışkanlara ilave olarak kullanıcı tarafından akışkan tanımlanıp veri tabanına eklenebilmesi imkanı tanınmıştır. Bu durumda akışkanların yoğunluk ve ısı sığaları üçüncü derece bir polinomla ifade edilir. Isıl iletkenlik ikinci derece bir polinomla, viskozite ve buhar basınçları ise üstel fonksyonlar ile tanımlanır (Denklem 1-6).

$$\rho = \sum_{i=0}^{n} C_{\rho,i} T^i \quad \text{Denklem 1}$$

$$c = \sum_{i=0}^{n} C_{c,i} T^i \quad \text{Denklem 2}$$

$$u = \sum_{i=0}^{n} \frac{1}{n+1} C_{c,i} \left(T^{i+1} - T_{ref}^{i+1}\right) \quad \text{Denklem 3}$$

$$s = C_{c,0} ln\left(\frac{T}{T_{ref}}\right) + \sum_{i=0}^{i-1} \frac{1}{i+1} C_{c,i+1} \left(T^{i+1} - T_{ref}^{i+1}\right) \quad \text{Denklem 4}$$

$$k = \sum_{i=0}^{n} C_{k,i} T^i \quad \text{Denklem 5}$$

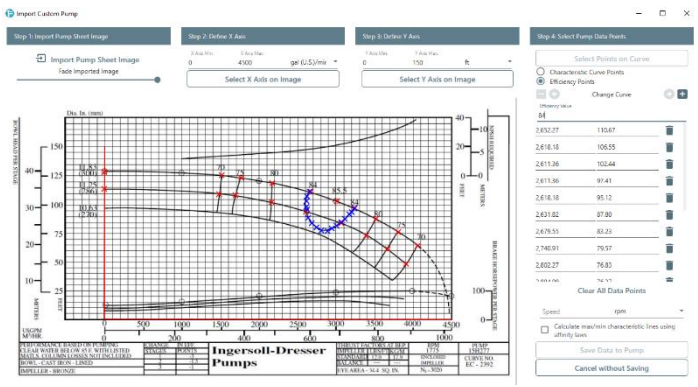
$$\mu = C_{\mu,0} exp\left(\frac{C_{\mu,1}}{T}\right) \quad \text{Denklem 6}$$

KOMPONENT VERİTABANI

Programda standart komponentlerin örneğin borular veya pompalar v.b. saklanması ve gerektiğinde kullanıcı tarafından genişletilebilmesi için bir SQL veritabanı bulunmaktadır. Şekil 12'de üreticinin sağladığı bir pompa eğirisinin kullanıcı tarafında sayısallaştırılıp veritabanına eklenmesi gösterilmektedir.

Veritabanı .NET platformundaki nesne ilişkisel eşleştirme araçlarından birisi olan Entity Framework teknolojisi ile oluşturulmaktadır. Bu teknik nesneye yönelik programlama ile veritabanı arasındaki ilişkiyi kurar. Böylece yazılımcı veritabanı ve SQL konutlarına ihtiyaç duymadan yazılım geliştirebilir. Entity Framework ile veritabanı geliştirmenin birkaç yolu

bulunmaktadır. Bu çalışmada ise "Önce Kod" yaklaşımı kullanılmıştır. Bu yaklaşımda sınıflar ve eşleştirme kodları yazılımcı tarafından oluşturulur ve daha sonra veri tabanı bu sınıflardan türetilir. Program içerisinde veritabanını sorgulamak için ise LINQ (language integrated query) yapısı kullanılır.



Şekil 12. Bir Pompa Eğrisinin Kullanıcı Tarafından Dijitize Edilip Veritabanına Eklenmesi

MATEMATİK MODELLEME

Model tabanlı tasarım yazılımının temel görevi kullanıcı tarafından grafik arayüzde kütüphanelerden sürükle bırak yaklaşımıyla tanımlanan bir modeli çekirdek katmanında matematiksel bir modele (diferansiyel-cebirsel denklem takımına) dönüştürmektir. Denklem 7 ile ifade edilen bu sisteminin bilgisayarda sayısal yöntemlerle çözülebilmesi için ise zaman koordinatı boyunca ayrıklaştırma yapılıp başlangıç zamanından benzetim süresinin sonuna kadar tümlev alınmalıdır.

$$\frac{dy}{dt} = f(t,y) \qquad \text{Denklem 7}$$

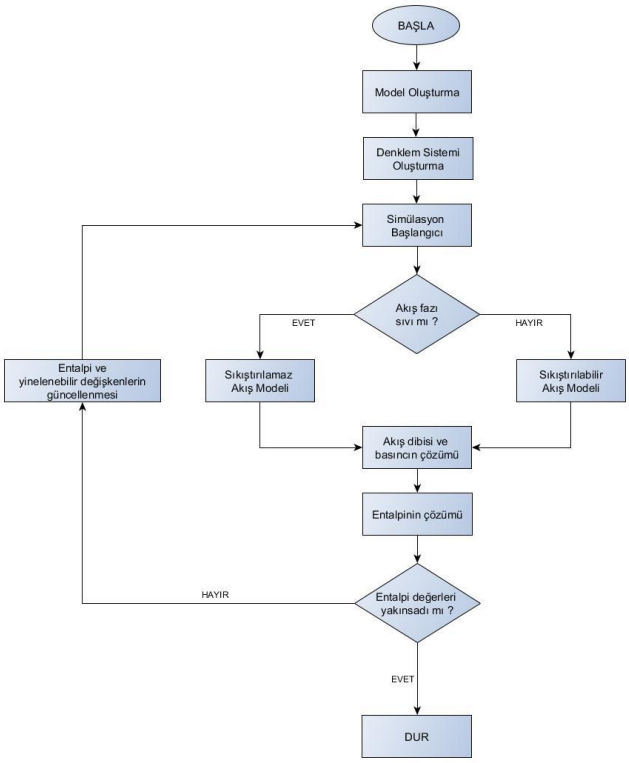
$$g(t,y) = 0$$

Örtülü Euler algoritmasına göre zaman içerisinde yapılan ayrıklaştırma ise Denklem 8'de verilmektedir. Burada mevcut zaman adımı k ile bir sonraki zaman adımı ise (k+1) ile temsil edilmektedir. Görüleceği üzere (k+1). Zaman adımındaki çözümü elde etmek için doğrusal olmayan bir kök bulma problemi çözmek gerekmektedir. Programda kök bulma algoritması olarak kullanıcı tercihine bağlı olarak Newton-Raphson, sönümlenmiş Newton-Raphson veya homotopi süreklilik algoritması kullanılmaktadır.

Sürekli rejimdeki bir problemde ise dinamik terimler (zamana bağlı türevler) bulunmamaktadır, yani başka bir deyişle sistem denge halindedir. Bu durumda çözüm esasında bir kök bulma problemine sadeleşmektedir. Akışın da probleme dahil olması ve akış yönündeki belirsizlikler özel bir takım yöntemlerle (akış değişkenleri kullanılarak) ele alınmaktadır. Sürekli rejimdeki bir probleme ilişkin çözüm algoritması temel adımları ile Şekil 13'te gösterilmektedir. Burada "akış değişkenleri" olarak ele alınan entalpi ve konstantrasyonun ayrıca ele alındığı görülmektedir ki bu sayede akışın yönü kolayca doğru hesaplanabilmektedir.

$$\frac{y^{k+1} - y^k}{\Delta t} = f(t^{k+1}, y^{k+1}) \qquad \text{Denklem 8}$$

$$g\left(t^{k+1}, y^{k+1}\right) = 0$$



Şekil 13. Sürekli Rejimdeki Model Çözüm Algoritması

Geliştirilen program blok diyagramı olarak ifade edilen bir modeli arka-planda bir diferansiyel cebirsel denklem takımına çevirip çözmektedir. Zamana bağlı çözüm için geriye dönük fark algoritmaları (BDF) kullanılmaktadır. Çözüm algoritmasının akış şeması Şekil 14'te gösterilmektedir. Sabit adımlı algoritmaların yanısıra adaptif adımlı algoritmalar da kodlanmıştır.

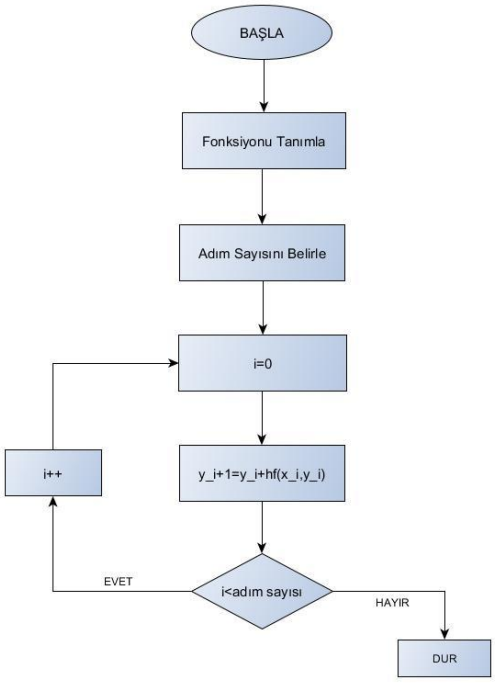
Yazılımda hesaplamalar birimli olarak yapılmaktadır. Her bir parametrenin, sonucun veya durum değişkenin birimi vardır. Bu maksatla UnitsNet kütüphanesi (17) yazılıma entegre edilmiştir. Matris vektör işlemleri v.b. matematiksel işlemler için ise yazılım Math.NET kütüphanesini (18) kullanmaktadır. Bahsi geçen kütüphaneler açık kaynaklı ve MIT lisanslı yani kullanımı tamamen serbest kütüphanelerdir.

DOĞRULAMA ÖRNEKLERİ

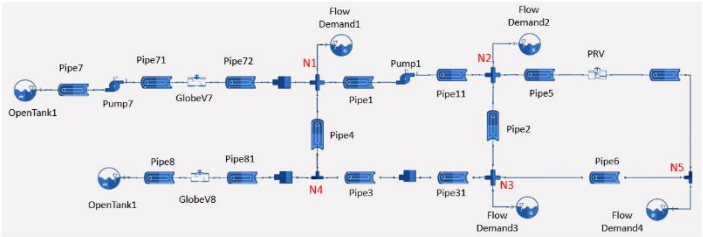
Bu bölümde çeşitli alanlardan doğrulama örnekleri ele alınmaktadır. Böylece programın yaptığı çözümler literatürdeki verilerle, teorik değerlerle ve diğer programların çıktıları ile karşılaştırılacaktır.

Basınç Regülatörü İçeren Dokuz Borulu Bir Akış Ağı

Bir akış ağında dokuz adet birbirine bağlı boru iki rezervuardan gelen suyu iki adet çıkış talep noktasına iletmektedirler. Akış ağında iki pompa, iki glob vana, ve bir akış-ölçer bulunmaktadır. Akış ağında iki açık bir kapalı döngü bulunur. Sekiz numaralı düğümdeki basınç regülatörü (PRV) vasıtasıyla akış 14,58 psig basınca ayarlanmaktadır. (149 m basma yüksekliğine eşdeğer). Problemin blok diyagramı Şekil 15'te gösterilmektedir. Akışkan 50° F sıcaklıkta sudur. Problem (19) numaralı referanstan uyarlanmıştır.



Şekil 14. Zamana Bağlı Bir Problemin Çözüm Algoritması



Şekil 15. Dokuz Borulu Akış Ağı Blok Diyagramı

Sonuçların karşılaştırılması Tablo 1'de sunulmuştur. Geliştirilen programın literatür ile uyumlu sonuçlar verdiği görülmektedir.

Tablo 1. Programın Ürettiği Sonuçların Literatürdekilerle Karşılaştırması

		Literatür	FlowNetMaster
Boru1.Port2	Debi, m^3/s	-0,1125	-0,1125
Boru2.Port2	Debi, m^3/s	0,0018	0,0027
Boru3.Port2	Debi, m^3/s	-0,1175	-0,1175
Boru4.Port2	Debi, m^3/s	0,0792	0,0799
Boru5.Port2	Debi, m^3/s	-0,0343	-0,0297
Boru6.Port2	Debi, m^3/s	-0,0657	0,070
Boru7.Port2	Debi, m^3/s	-0,0633	0,0625
Boru8.Port2	Debi, m^3/s	-0,1967	-0,1974
Boru9.Port2	Debi, m^3/s	-0,0343	-0,0297
Pompa7	Basma Yüksekliği, m	6,18	6,38
Pompa1	Basma Yüksekliği, m	3,58	3,58
PRV	HGL, m	149	149

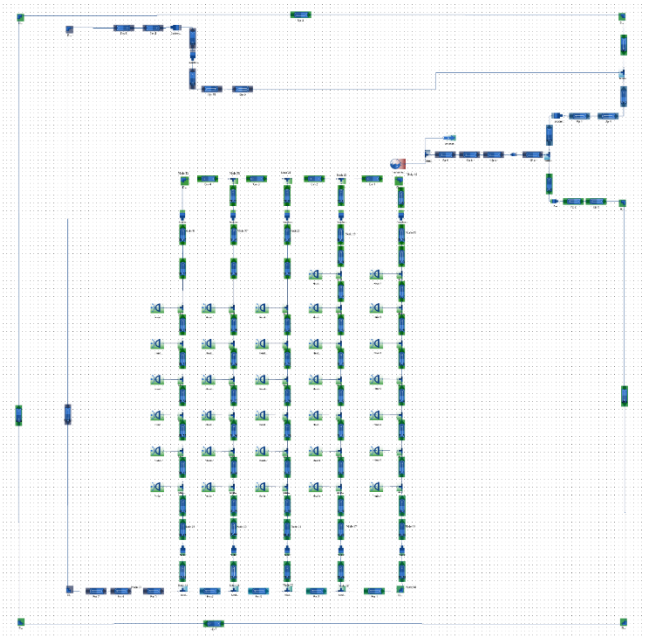
Bir İmalat Tesisisin Yangın Güvenliği Sistemi

Bir diğer doğrulama örneği de bir imalat tesisine ait yangın güvenlik sistemidir. Ana şebekeden gelen su bir dizi branşa ayrıldıktan sonra sprinklerler ile ortama deşarj edilmektedir. Sistemin blok diyagramı Şekil 16'da gösterilmektedir. Sistemde toplam 31 adet sprinkler bulunmaktadır. Sonuçlar Tablo 2'de bir ticari paket programın verdiği sonuçlar ile karşılaştırılmaktadır. Geliştirdiğimiz yazılım ile ticari paket programın (SprinkCAD) sonuçları uyumludur. Debilerdeki ufak farklar SprinkCAD'in basınç düşümü hesabı için nispeten daha basit olan Colebrook-White formülünü, FlowNetMaster'in ise Darcy-Weisbach formülünü kullanmasından kaynaklanmaktadır ve bu durum göz önüne alındığında ise hesaplama farklılıkları beklenilen mertebededir.

Tablo 2. Elde Edilen Sonuçların Ticari Bir Paket Programın Verdiği Sonuçlarla Karşılaştırılması

#	Tip	Parametre Değeri	Deşarj (lpm)	
			Ticari Paket Program	FlowNetMaster
s1	Kaynak	1893,3	1893,3	1893,3
h1	Sprinkler	160	156,8	171,3
h2	Sprinkler	160	142,9	155,7
h3	Sprinkler	160	133,0	144,8
h4	Sprinkler	160	126,6	138,1
h5	Sprinkler	160	123,2	134,6
h6	Sprinkler	160	121,9	133,4
h7	Sprinkler	160	121,8	133,2
h8	Sprinkler	160	155,5	170,1
h9	Sprinkler	160	141,5	154,3
h10	Sprinkler	160	131,5	143,4
h11	Sprinkler	160	125,0	136,5
h12	Sprinkler	160	121,4	132,9
h13	Sprinkler	160	120,0	131,5
h14	Sprinkler	160	119,9	131,4
h15	Sprinkler	160	152,7	166,5

h16	Sprinkler	160	139,1	151,3
h17	Sprinkler	160	129,5	140,7
h18	Sprinkler	160	123,2	134,1
h19	Sprinkler	160	119,9	130,7
h20	Sprinkler	160	118,6	129,5
h21	Sprinkler	160	154,2	167,9
h22	Sprinkler	160	140,7	152,8
h23	Sprinkler	160	131,1	142,3
h24	Sprinkler	160	125,1	135,9
h25	Sprinkler	160	121,8	132,6
h26	Sprinkler	160	120,7	131,4
h27	Sprinkler	160	155,7	169,4
h28	Sprinkler	160	142,2	154,3
h29	Sprinkler	160	132,8	143,9
h30	Sprinkler	160	126,9	137,6
h31	Sprinkler	160	123,8	134,4
h32	Sprinkler	160	122,7	133,3
Toplam Sprinkler Deşarjı			6155,0	6492,9
Ortalama Sprinkler Deşarjı			131,9	143,7
Balanssızlık			24,9	27,6



Şekil 16. Bir İmalat Tesisinin Sprinkler Sistemi

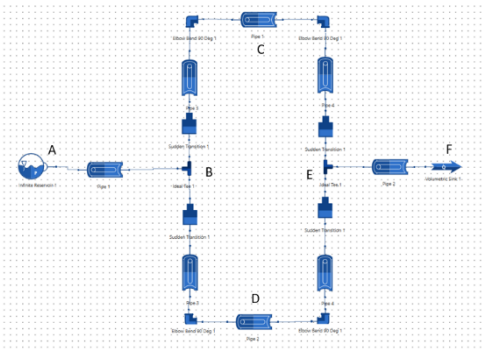
Metan Gazı Paralel Boru Hattı

Bir gaz boru hattı, 100 MMSCFD akış debisinde çalışacak şekilde tasarlanan Şekil 17'de gösterildiği gibi iki paralel borudan oluşmaktadır. AB segmenti 12 kara mili uzunluğunda olup, iç çapı 15 ½ " olan borulardan oluşur. BCE kolu ise 24 kara mili uzunluğunda ve 13 ½ " iç çaplı borulara sahiptir. BDE kolu ise yine 12 kara mili uzunluğunda ve 12 ¼" iç çaplı borulardan oluşmaktadır. EF kısmı 20 kara mili uzunluğunda olup, iç çapı 15 ½" olan bir borudan oluşur. 0,6'lık bir gaz yoğunluğu, hava yoğunluğu oranı varsayılarak, F noktasındaki çıkış basıncını ve paralel boru branşlarının başlangıcındaki ve sonundaki basınçların hesaplanılması istenmektedir.

A sınır noktasında yer alan kaynakta basınç ve sıcaklık aşağıda verildiği gibidir.

- P_A= 1200 Psig;
- T_A= 80 Fahrenheit

Basınç düşümü hesabı için genel akış denklemi (general Flow equation) kullanılmıştır.



Şekil 17. Metan Gazı Paralel Boru Hattı Modeli Blok Diyagramı

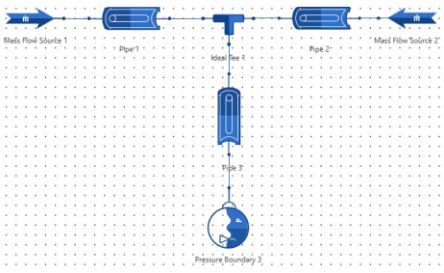
Tablo 3. FlowNetMaster Sonuçlarının Bir Ticari Paket Program Sonuçları ile Karşılaştırılması

	B Noktası Basıncı (psig)	E Noktası Basıncı (psig)	F Noktası Basıncı (psig)
FlowNetMaster	1175.2	1153	1101.4
PipeFlow Expert Çözümü	1166.6	1130.9	1071.12

Görüleceği üzere ticari paket program PipeFlow Expert çözümü ile FlowNetMaster'in çözümü birbirleri ile neredeyse birebir uyumludur.

Nemli Hava Akışlarının Karışması

Kütüphanede nemli hava da yer almaktadır. Nemli hava çözülürken havanın nem oranı da aynı "toplam entalpi" gibi bir akış değişkeni olarak çözülmektedir.



Şekil 18. Nemli Hava Akımlarının Karışımları Problemi Blok Diyagramı

Örnek problemde iki nemli hava akışının karışımı çözülmektedir. Doğrulama örneğine ait blok diyagramı Şekil

18'de gösterilmektedir. Kaynak 1'in kütle debisi 1,5 kg/s, sıcaklığı 288 K, bağıl nem oranı ise %80'dir. Kaynak 2'nin ise kütle debisi 0,5 kg/s, sıcaklığı 288 K, bağıl nem oranı ise %20'dir. İki akış karışmakta ve bir basınç sınırına doğru akmaktadır. Tablo 4'te görüldüğü üzere yazılımın verdiği sonuçlar teorik değerlerle uyumludur.

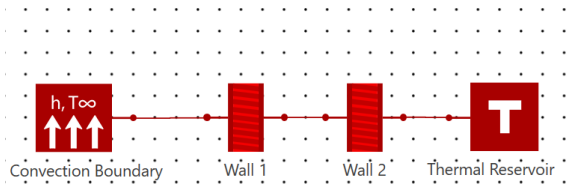
Tablo 4. FlowNetMaster Sonuçları ile Teorik Sonuçların Karşılaştırılması

	FlowNetMaster	Teorik Değer
Basınç Sınırı Kütle Debisi, kg/s	2,0	2,0
Basınç Sınırı Sıcaklığı, K	288	288
Basınç Sınırı Nem Bağıl Oranı	%35	%35

Kompozit Duvardan Isı Geçişi

Şekil 19'da gösterildiği üzere bir kompozit duvar (iki farklı malzemeden yapılmış) sol tarafında taşınım sınır şartına sağ tarafında ise sabit sıcaklık sınır şartına maruz kalmaktadır. Probleme ilişkin parametreler aşağıda verilmiştir;

- $T_{akışkan}$ = 350 K
- $h_{taşınım}$ = 10 W/m^2.K
- $k_{wall,1}$ = 2 W/m.K
- $k_{wall,2}$ = 3 W/m.K
- $A_{wall,1}$ = 0.5 m^2
- $A_{wall,2}$ = 0.5 m^2
- $L_{wall,1}$ = 0.3 m
- $L_{wall,2}$ = 0.3 m
- $T_{rezervuar}$= 300 K



Şekil 19. Kompozit Duvar Isı Geçişi Örneği

Bu verilere göre ısı geçişi miktarının ve duvar yüzeylerindeki sıcaklıkların hesaplanması gerekmektedir.

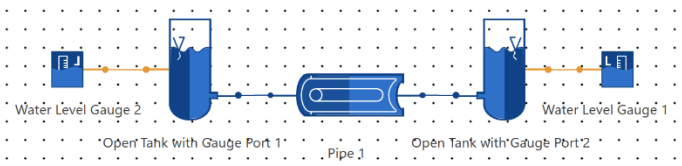
Hesaplama sonuçları Tablo 5'te özetlenmektedir. Görüleceği üzere FlowNetMaster'in çözümleri teorik değerler ile tamamen uyumludur.

Tablo 5. Teorik Değerler ile FlowNetMaster Çözümünün Karşılaştırılması

	FlowNetMaster	Teorik Değer
Taşınım Sınır Şartı Isı Geçişi, W	71,42	71,41
Taşınım Sınır Şartı Sıcaklık, K	335,71	335,72
Duvar 1 Port2 Isı Geçişi, W	71,42	71,41
Duvar 1 Port2 Sıcaklık, K	314,29	314,30
Duvar 2 Port2 Isı Geçişi, W	71,42	71,41
Duvar 2 Port2 Sıcaklık, K	300	300

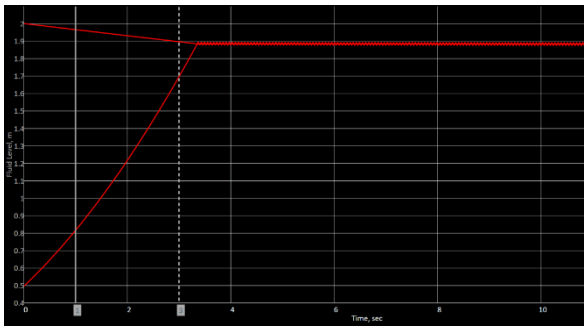
Bileşik Kaplarda Zamana Bağlı Akış

Şekil 20'de birbirlerine bir boru vasıtası ile bağlı iki tank gösterilmektedir. Tanklardaki su seviyeleri ve/veya tankların yerden yükseklikleri başlangıçta t=0 farklıdır. İki tank boru vasıtasıyla birbirleriyle İletişim halinde olduklarından yükseklik farkından ötürü yüksekten alçağa doğru bir akış başlar ve su seviyelerinin yerden yükseklikleri (tanktaki seviyeleri değil) eşitlenene kadar akış devam eder.

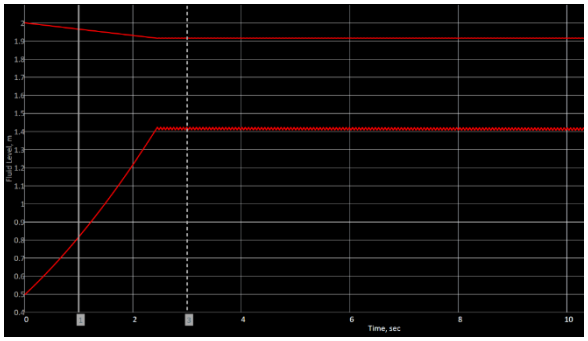


Şekil 20. Bileşik Kaplar Modelinin Blok Diyagramı

Şekil 21'de görüleceği üzere aynı yerden yükseklikteki tanklar arasında tank seviyelerinin eşitlenmesi görülmektedir.



Şekil 21. 0.8 m ve 0.2 m çapta olan silindirik tanklardaki akışkan seviyelerin zamana bağlı dengeye ulaşması



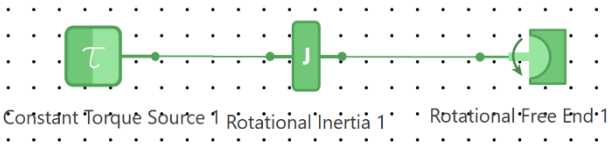
Şekil 22. Çapı 0.8 m ve 0.2 m ve yükseklik 0 m ve 0.5 m olan tanklardaki akışkan seviyelerin zamana bağlı dengeye ulaşması

Şekil 22'de ise yerden yükseklikleri birbirinden 0,5 m farklı olan iki tanktaki tanka göre su seviyelerinin değişimi görülmektedir. Simülasyon sürekli rejimde denge noktasına ulaştığında iki tanktaki su kolonlarının yerden yükseklikleri aynı olmaktadır. Bu fiziksel sonuç "bileşik kaplar yasasının" bir gereğidir.

Eylemsizliği Olan Bir Şarftın Sabit Tork ile İvmelenmesi

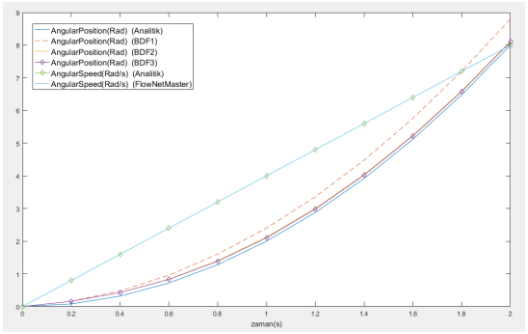
Mekanik sistemlere örnek olarak ele alınan problem bir şaftın uygulanan sabit bir tork ile ivmelenmesidir. Probleme ilişkin FlowNetMaster kullanıcı grafik arayüzünde oluşturulan blok diyagram Şekil 23'te gösterilmektedir. Model parametreleri ise aşağıda belirtildiği gibidir.

- Tork = 20 Newton.metre
- Eylemsizlik = 5 kg.m^2



Şekil 23. Tork ve Şaft Probleminin Blok Diyagramı

Newton mekaniğinden gelen analitik çözüm açısal hızın lineer olarak artması, konumun ise parabolik bir eğri takip etmesidir. Şekil 24'te gösterildiği üzere analitik çözüm ile sayısal çözüm arasında bire bir tutarlılık bulunmaktadır. Bu örnek aynı zamanda kodlanan zamana bağlı çözüm algoritmasının doğrulunu test etmek için kullanılan otomatik test kütüphanemizdeki testlerden birisini de oluşturmaktadır.



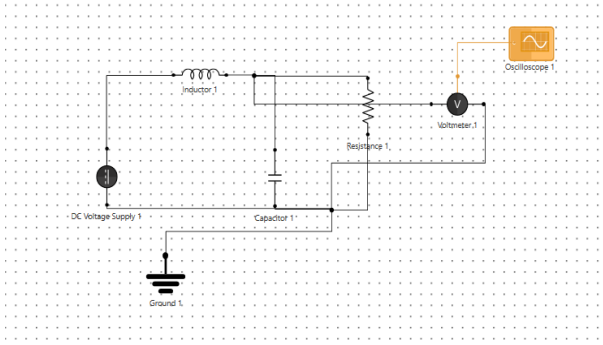
Şekil 24. FlowNetMaster Çözümü ile Analitik Çözümün Karşılaştırılması

RLC Devresinin Zamana Bağlı Çözümü

RLC (direnç-indüktans-kapasitans) devresi temel elektrik devrelerinden birisidir ve bu dinamik problem analitik olarak kolayca çözülebilmektedir. RLC problemine ait model parametreleri ise aşağıda belirtilmektedir. Bu problem de CI/CD iş akışındaki doğrulama testleri arasında yer almaktadır.

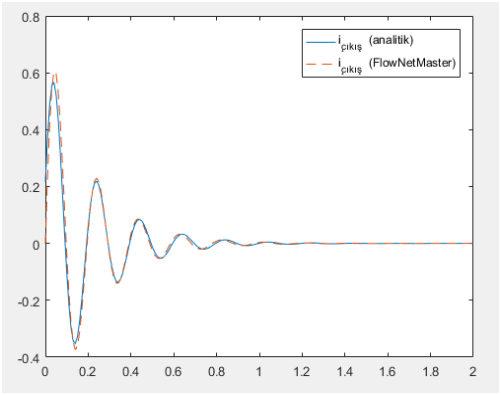
- V = 24 Volt
- L = 1 Henry
- R = 100 Ω
- C = 0.001 Farad
- t = 0 : 2.0 : 0.01 saniye

Probleme ilişkin FlowNetMaster blok diyagramı ise Şekil 25'te gösterilmektedir.



Şekil 25. RLC Devre Blok Diyagramı

Şekil 26'da ise geliştirilemn yazılımın verdiği çözüm ile analitik çözüm birbirleri ile karşılaştırılmaktadır. Görüleceği üzere elde edilen sayısal çözüm ile analitik çözüm birbirleri ile birebir uyumludur.



Şekil 26. FlowNetMaster Çözümü ile Analitik Çözümün Karşılaştırılması

Değişken Doğru Akım Gerilim Kaynaklı RC Devresi Şeması

Değişken doğru akım (DC) kaynaklı RC devre şeması Şekil 27'de gösterilmektedir. Model parametreleri ise aşağıda belirtildiği gibidir.

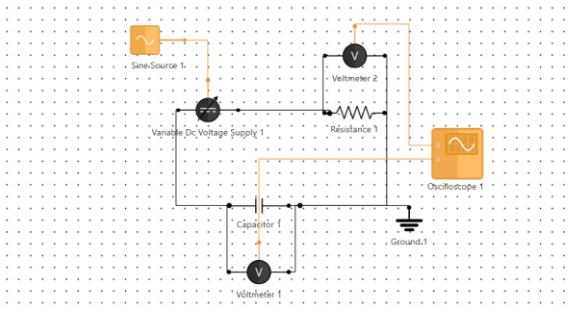
- R = 50 Ω
- C = 0.1 miliFarad
- $V_s = V_0 \sin(2\pi ft)$
- V_0 = 100 Volt
- f = 60 Hertz
- t = 0 ila 0.1saniye arası, 1 milisaniye zaman adımı

Devre cevabının analitik çözümü Denklem 9'da verilmektedir.

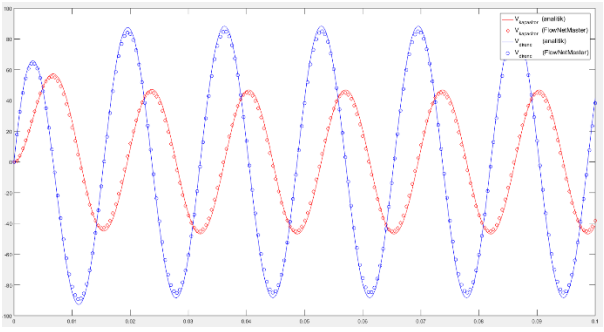
$$V_c(t) = A\sin 2\pi f + B\cos 2\pi f + Ce^{-t/\tau}$$
$$A = \frac{V_0}{1+(2\pi f\tau)^2}\,, \qquad B = -A2\pi f\tau\,, \qquad C = -B \quad , \quad \tau = RC$$

Denklem 9

FlowNetMaster çözümü için BDF-1 (örtülü Euler) algoritması kullanılmıştır. Şekil 28'de FlowNetMaster programının hesapladığı sonuç ile analitik çözüm beraber gösterilmektedir. Programdan elde edilen çözümle analitik çözüm birbirlerinin üzerine tamamen oturmaktadır.



Şekil 27. Değişken Doğru Akım Gerilim Kaynaklı RC Devresinin Şeması



Şekil 28. FlowNetMaster Çözümünün Analitik Çözümle Karşılaştırılması

Doğal Cevap RC Devresi

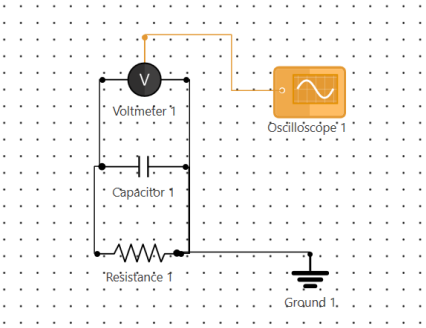
Bir diğer doğrulama örneği ise doğal cevap RC devresidir. Devre şeması Şekil 29'da gösterilmektedir. Probleme ilişkin parametreler ise aşağıda sıralanmıştır.

- V_{c0} = 100 V
- R = 5 Ω
- t = 0 ila 15 saniye arası, 0,5 saniye zaman adımı

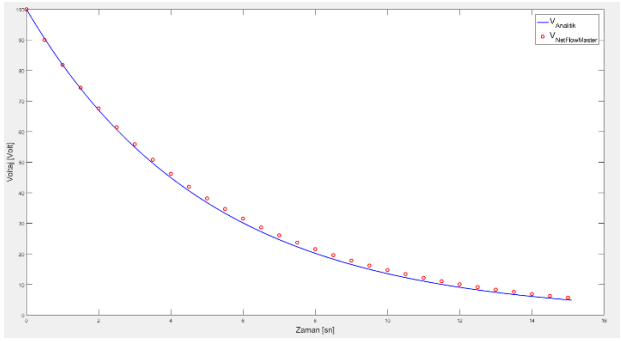
Şekil 30'da FlowNetMaster'in çözümü ile analitik çözüm karşılaştırılmaktadır. Görüleceği gibi bir evvelki doğrulama örneğinde olduğu gibi FlowNetMaster'in çözümü analitik çözüm ile birebir örtüşmektedir.

Çift Borulu Isı Değiştiricisi

5.000 kg/saat debideki soğuk su eşanjöre giriş sıcaklığı 132.5°C olan sıcak su vasıtasıyla, 20°C sıcaklıktan 35°C sıcaklığa kadar ısıtılmaktadır. Sıcak suyun eşanjörden çıkış sıcaklığının giriş sıcaklığının 15°C altında olabileceği belirtilmektedir. Bu iş için firkete (hairpin) konfigürasyonunda seri bağlanan iki adet çift borulu ısı değiştiricisi kullanılmaktadır. Dıştaki borunun çapı 3", içteki borunun çapı ise 2" olarak belirtilmektedir. Sıcak su iç taraftan, soğuk su ise dıştan akmaktadır. İç ve dış taraflardaki kirlenme faktörleri sırasıyla $R_{f,iç}$ = 0.000176 m².K /W, $R_{f,dış}$ = 0.000352 m².K / W olarak belirtilmektedir. Borular ısı iletim katsayısı k = 54 W/m.K olan karbon çeliğinden imal edilmişlerdir. Isı değiştiricisi, ısı kayıplarına karşı yalıtılmıştır. Problem (20) numaralı referanstan adapte edilmiştir. Probleme ilişkin FlowNetMaster blok diyagramı Şekil 31'de gösterilmektedir. Sistem sürekli rejimde çalışmaktadır.



Şekil 29. Doğal Cevap RC Devresi Blok Diyagramı



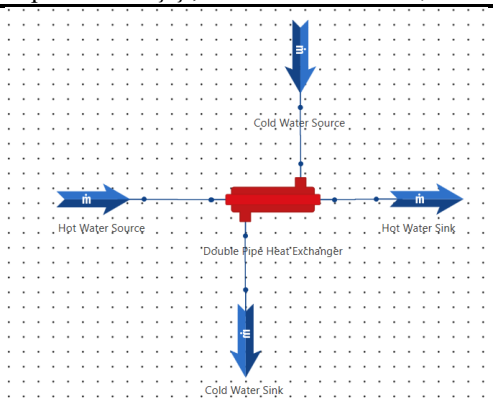
Şekil 30. FlowNetMaster Çözümü ile Analitik Çözümün Karşılaştırılması

FlowNetMaster'in çözümleri ile literatürdeki (20) çözümlerin birbirleri ile örtüştüğü Tablo 6'da görülmektedir.

Tablo 6. FlowNetMaster sonuçları ve literatür verilerinin karşılaştırma tablosu

	FlowNetMaster	Literatür
İç Boru Reynolds Sayısı	158092	159343
İç Boru Basınç Kaybı, Pa	461,68	460,1
İç Boru Nusselt Sayısı	376,5	375,3
İç Boru Taşınım Katsayısı, W/m².K	4898,4	4911,0
Dış Boru Reynolds Sayısı	12768	15201
Dış Boru Basınç Kaybı, Pa	3094	2876,4

Dış Boru Nusselt Sayısı	78,9	89,0
Dış Boru Taşınım Katsayısı, $W/m^2.K$	1170,4	1345,0
Toplam Isı Geçişi Katsayısı, $W/m^2.K$	581	622
Toplam Isı Geçişi, kW	87,0	87,1



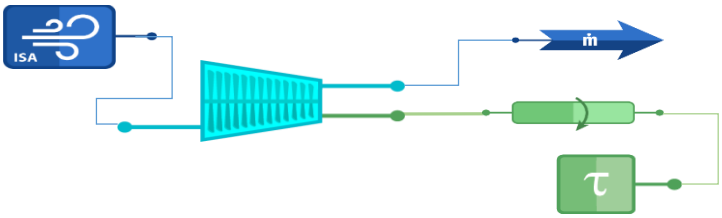
Şekil 31. Çift Borulu Eşanjör Modeli Blok Diyagramı

Eksenel Kompresör Örneği

Şekil 32'de eksenel kompresör doğrulama örneğine ait FlowNetMaster blok diyagramı görülmektedir. Problemin analitik çözümü mümkün olmadığı için bu problem MATLAB SIMULINK ortamı için NASA tarafından geliştilen açık kaynak kodlu TMATS kütüphanesi (21) ile sayısal olarak çözülmüş ve çözümler FlowNetMaster çözümlerini doğrulamak için kullanılmıştır. TMATS blok diyagramı Şekil 34'te gösterilmektedir.

Doğrulama örneğinde J-85 motoruna ait kompresörün haritası kullanılmıştır. Probleme ait girdiler aşağıdali gibidir.

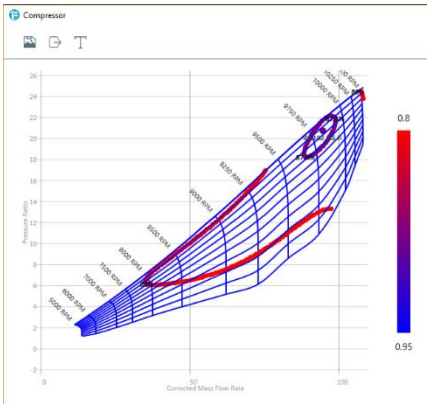
- Toplam Sıcaklık = 298 Kelvin
- Toplam Basınç = 101,3 kiloPascal
- Hava Debisi = 100 kg/s
- Tork = 51.946 kN.m



Şekil 32. Kompresör Doğrulama Modeli Blok Diyagramı

Şekil 33'te problemdeki kompresöre ait harita gösterilmektedir. Haritada yer alan temel bilgiler şunlardır;

- Harita ölçekleme katsayıları
- Düzeltilmiş hız vektörü
- Rline vektörü
- Düzeltilmiş kütle debisi matrisi
- Verim matrisi
- Basınç oranı matrisi
- Surge eğrisi düzeltilmiş kütle debileri
- Surge eğrisi basınç oranları



Şekil 33. Kompresör Haritası

Şekil 33'teki haritaya ilişkin bilgiler aşağıda Harita 1'de verilmektedir. Geliştirilen FlowNetMaster programbu dosyayı okuyup Şekil 33'teki haritayı çizmektedir.

```
correctedSpeedUnit      "RPM"
correctedMassFlowRateUnit   "kg/s"

correctedMassFlowRateScaler 0.495342
pressureRatioScaler 0.863640
efficiencyScaler  0.997653
correctedSpeedScaler 10000

correctedSpeedVector
0.500 0.600 0.700 0.750 0.800 0.850 0.900 0.925
0.950 0.975 1.000 1.025 1.050

RlineVector
1.000 1.200 1.400 1.600 1.800 2.000
2.200 2.400 2.600 2.800 3.000

correctedMassFlowRateMatrix
22.7411 24.0487 25.1548 26.0615 26.7738 27.2992
27.6470 27.8286 27.8634 27.8634 27.8634
31.7548 33.1181 34.2670 35.2054 35.9397 36.4783
36.8308 37.0085 37.0362 37.0362 37.0362
46.1066 47.4088 48.5066 49.4046 50.1096 50.6291
50.9717 51.1469 51.1757 51.1757 51.1757
56.7268 58.0480 59.1608 60.0704 60.7837 61.3084
61.6527 61.8260 61.8517 61.8517 61.8517
70.1448 71.5163 72.6688 73.6088 74.3429 74.8795
75.2269 75.3943 75.4134 75.4134 75.4134
89.3764 90.9746 92.3098 93.3900 94.2232 94.8199
95.1897 95.3442 95.3504 95.3504 95.3504
118.0620 120.1207 121.8253 123.1867 124.2166 124.9292
125.3385 125.4609 125.4609 125.4609 125.4609
138.5093 140.8966 142.8639 144.4238 145.5916 146.3836
146.8174 146.9192 146.9192 146.9192 146.9192
160.6243 162.5676 164.1805 165.4722 166.4536 167.1370
167.5334 167.6563 167.6563 167.6563 167.6563
```

181.7993 183.4993 184.9150 186.0545 186.9260 187.5389
187.9029 188.0273 188.0271 188.0271 188.0271
202.6315 203.5858 204.3958 205.0661 205.5998 206.0000
206.2702 206.4145 206.4418 206.4418 206.4418
209.9986 210.5917 211.1029 211.5321 211.8825 212.1554
212.3516 212.4735 212.5220 212.5227 212.5227
216.6847 217.0279 217.3287 217.5860 217.8015 217.9767
218.1106 218.2041 218.2586 218.2739 218.2739

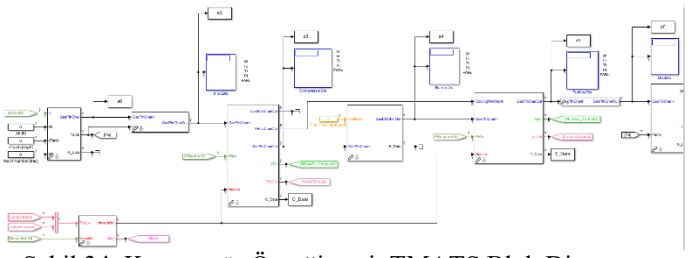
efficiencyMatrix
0.6753 0.6913 0.7016 0.7050 0.7004 0.6864 0.6570
0.6044 0.5236 0.4075 0.2467
0.6953 0.7094 0.7184 0.7214 0.7176 0.7058 0.6812
0.6378 0.5717 0.4783 0.3512
0.7248 0.7359 0.7429 0.7452 0.7424 0.7335 0.7154
0.6838 0.6366 0.5713 0.4848
0.7427 0.7533 0.7600 0.7627 0.7606 0.7533 0.7379
0.7108 0.6703 0.6147 0.5414
0.7634 0.7736 0.7804 0.7834 0.7822 0.7762 0.7630
0.7394 0.7041 0.6556 0.5920
0.7891 0.8008 0.8090 0.8134 0.8136 0.8092 0.7974
0.7754 0.7417 0.6950 0.6335
0.8139 0.8280 0.8385 0.8449 0.8469 0.8439 0.8333
0.8117 0.7779 0.7303 0.6671
0.8206 0.8356 0.8469 0.8541 0.8567 0.8544 0.8442
0.8229 0.7892 0.7416 0.6783
0.8403 0.8512 0.8593 0.8643 0.8660 0.8641 0.8566
0.8415 0.8179 0.7852 0.7423
0.8408 0.8492 0.8552 0.8588 0.8597 0.8578 0.8516
0.8394 0.8209 0.7954 0.7624
0.8470 0.8505 0.8529 0.8539 0.8536 0.8520 0.8483
0.8418 0.8324 0.8200 0.8043
0.8350 0.8364 0.8371 0.8370 0.8362 0.8346 0.8318
0.8275 0.8217 0.8141 0.8049
0.8202 0.8203 0.8201 0.8195 0.8185 0.8171 0.8152
0.8124 0.8088 0.8045 0.7992

pressureRatioMatrix
2.4769 2.4288 2.3620 2.2778 2.1774 2.0627
1.9284 1.7711 1.5958 1.4083 1.2146
3.4633 3.3778 3.2643 3.1248 2.9619 2.7787
2.5679 2.3253 2.0595 1.7802 1.4973
5.0821 4.9375 4.7554 4.5391 4.2923 4.0194
3.7106 3.3602 2.9800 2.5826 2.1813
6.3490 6.1658 5.9371 5.6667 5.3594 5.0204
4.6377 4.2042 3.7342 3.2431 2.7467
8.0021 7.7686 7.4792 7.1388 6.7532 6.3287
5.8504 5.3097 4.7237 4.1114 3.4919
10.4899 10.1976 9.8249 9.3786 8.8669 8.2989
7.6539 6.9201 6.1229 5.2899 4.4495
14.4564 14.0970 13.6074 12.9977 12.2808 11.4715
10.5377 9.4621 8.2878 7.0614 5.8306
17.4426 17.0500 16.4870 15.7661 14.9034 13.9183
12.7692 11.4347 9.9718 8.4425 6.9104
20.7403 20.2486 19.6093 18.8329 17.9324 16.9227
15.7626 14.4263 12.9562 11.3983 9.8011
23.8298 23.2601 22.5536 21.7200 20.7705 19.7178
18.5212 17.1524 15.6466 14.0424 12.3810
26.6962 26.0933 25.4175 24.6733 23.8656 22.9999
22.0495 20.9930 19.8436 18.6163 17.3267
27.6439 27.0687 26.4522 25.7969 25.1052 24.3798
23.6033 22.7614 21.8600 20.9058 19.9057
28.4663 27.9667 27.4460 26.9054 26.3460 25.7690
25.1640 24.5225 23.8472 23.1399 22.4038

surgeLineCorrectedMassFlowRate
22.7411 31.7548 46.1066 56.7268 70.1448 89.376
118.062 138.509 160.624 181.799 202.631 209.9986
216.684 218.2739

surgeLinePressureRatio
2.4769 3.4633 5.0821 6.3490 8.0021
10.4899 14.4564 17.4426 20.7403 23.8298
26.6962 27.6439 28.4663 28.6619

Harita 1. Kompresör Haritası ASCII Dosyası



Şekil 34. Kompresör Örneğine ait TMATS Blok Diyagramı

Tablo 7'de FlowNetMaster yazılımının verdiği sonuçlar ile Simulink TMATS kütüphanesinin verdiği sonuçlar karşılaştırılmaktadır. Görülmektedir ki iki yazılım da kompresör için aynı işletme noktasını (operating point) vermektedirler.

Tablo 7. FlowNetMaster ile TMATS Çözümlerinin Karşılaştırılması

	FlowNetMaster	TMATS
Düzeltilmiş Kütle Debisi (lb/s)	220	206
Düzeltilmiş Devir Sayısı (rpm)	10000	10000
Basınç Oranı	23,4	23,4
Verim	0,842	0,842
Giriş Toplam Basıncı (psi)	14,7	14,7
Çıkış Toplam Basıncı (psi)	345	344
Giriş Toplam Sıcaklık (R)	518,7	518,7
Çıkış Toplam Sıcaklık	1316	1316
Toplam Entalpi Farkı (BTU/lb)	197,1	197,1

Görüleceği üzeri FlowNetMaster ile TMATS'in çalışma noktası çözümleri birbirleriyle tam olarak uyumludur.

Eksenel Türbin Örneği

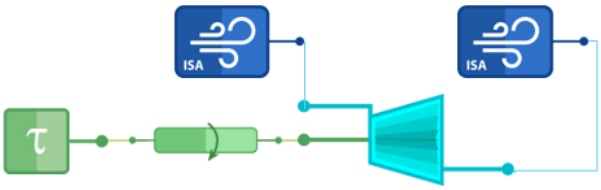
Benzer bir doğrulama çalışması bu sefer türbin örneği için yapılmıştır. Probleme ilişkin blok diyagramı Şekil 35'te türbin haritası ise Şekil 36 ve Harita 2'de verilmektedir. Sonuçlar ise kompresör örneğinde olduğu gibi TMATS çözümleri ile karşılaştırılmaktadır.

Probleme ilişkin veriler şu şekildedir;

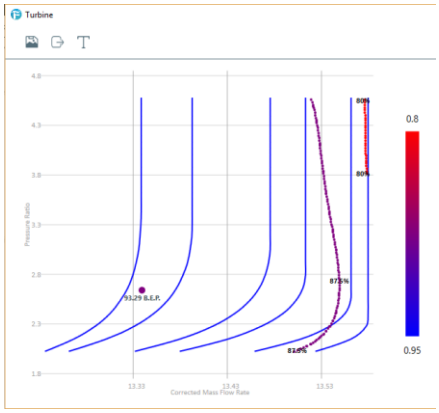
- Rakım = Deniz seviyesi
- Standart gün sıcaklık farkı = 0 K
- Kütle Akış Debisi= 100 kg / s
- Tork Talebi = 51.946 kN.m

Şekil 36'da probleme ilişkin türbin haritası (J-85 motoru için) gösterilmektedir. Türbin haritasında yer alan ana bilgiler şunlardır;

- Harita ölçekleme katsayıları
- Düzeltilmiş hız vektörü
- Basınç oranı vektörü
- Düzeltilmiş kütle debisi matrisi
- Verim matrisi



Şekil 35. Türbin Doğrulama Modeli Blok Diyagramı



Şekil 36. Türbin Haritası

```
correctedSpeedUnit      "RPM"
correctedMassFlowRateUnit   "kg/s"

correctedMassFlowRateScaler 0.444192
pressureRatioScaler 0.511630
efficiencyScaler 0.998914
correctedSpeedScaler 45.611020

correctedSpeedVector
60 70 80 90 100 110

pressureRatioVector
3.000 3.250 3.500 3.750 4.000 4.250 4.500
4.750 5.000 5.250 5.500 5.750 6.000 6.250
6.500 6.750 7.000 7.250 7.500 8.000

correctedMassFlowRateMatrix
30.446 30.533 30.568 30.572 30.572 30.572 30.572
30.572 30.572 30.572 30.572 30.572 30.572 30.572
30.572 30.572 30.572 30.572 30.572 30.572
30.299 30.413 30.480 30.516 30.529 30.530 30.530
30.530 30.530 30.530 30.530 30.530 30.530 30.530
30.530 30.530 30.530 30.530 30.530 30.530
30.120 30.239 30.317 30.368 30.398 30.415 30.421
30.421 30.421 30.421 30.421 30.421 30.421 30.421
30.421 30.421 30.421 30.421 30.421 30.421
30.014 30.124 30.201 30.253 30.288 30.311 30.325
30.333 30.337 30.337 30.337 30.337 30.337 30.337
30.337 30.337 30.337 30.337 30.337 30.337
29.856 29.948 30.013 30.059 30.091 30.113 30.128
30.139 30.145 30.149 30.150 30.150 30.150 30.150
30.150 30.150 30.150 30.150 30.150 30.150
29.799 29.870 29.920 29.955 29.979 29.997 30.009
30.017 30.023 30.026 30.028 30.029 30.029 30.029
30.029 30.029 30.029 30.029 30.029 30.029

efficiencyMatrix
0.8460 0.8405 0.8349 0.8296 0.8249 0.8206 0.8165
0.8127 0.8092 0.8060 0.8030 0.8002 0.7976 0.7953
0.7931 0.7911 0.7892 0.7875 0.7858 0.7826
0.8879 0.8842 0.8804 0.8769 0.8735 0.8701 0.8670
0.8640 0.8614 0.8590 0.8568 0.8548 0.8529 0.8511
0.8495 0.8479 0.8460 0.8436 0.8414 0.8373
0.9125 0.9111 0.9096 0.9078 0.9055 0.9034 0.9014
0.8995 0.8979 0.8964 0.8950 0.8936 0.8924 0.8903
0.8877 0.8853 0.8830 0.8808 0.8787 0.8749
0.9228 0.9242 0.9250 0.9247 0.9240 0.9232 0.9224
0.9217 0.9210 0.9203 0.9197 0.9188 0.9162 0.9137
0.9113 0.9091 0.9070 0.9050 0.9031 0.8980
0.9215 0.9258 0.9289 0.9304 0.9313 0.9319 0.9323
0.9326 0.9328 0.9329 0.9330 0.9311 0.9288 0.9266
0.9245 0.9225 0.9206 0.9188 0.9161 0.9107
0.9106 0.9176 0.9232 0.9267 0.9292 0.9312 0.9327
0.9340 0.9351 0.9361 0.9369 0.9349 0.9329 0.9311
0.9293 0.9276 0.9259 0.9239 0.9212 0.9161
```

Harita 2. Kompresör Haritası ASCII Dosyası

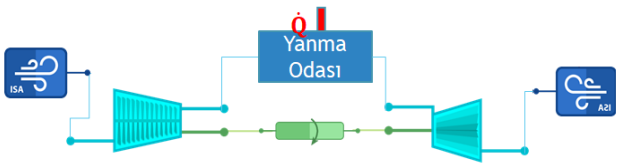
Tablo 8. Türbin Modeli Sonuçlarının Karşılaştırması

	FlowNetMaster	TMATS
Düzeltilmiş Kütle Debisi, lbm/s	227,8	227,1
Düzeltilmiş Devir Sayısı, devir/dakika	4105	4106
Basınç Oranı	3,0	3,0
Verim	0,962	0,962
Giriş Toplam Basınç, psi	273,6	273,6
Çıkış Toplam Basınç, psi	89,82	89,82
Giriş Toplam Sıcaklık, R	3078	3078
Çıkış Toplam Sıcaklık, R	2326	2453
Giriş Toplam Entalpi, BTU/lbm	847,7	847,7
Çıkış Toplam Entalpi, BTU/lbm	596,8	656,3

Tablo 8'den anlaşılacağı üzere FlowNetMaster ile TMATS'in çalışma noktası çözümleri birbirleriyle tam olarak uyumludur.

Türbin Kompresör Eşleşmesi

Yazılım geliştirme sürecinde türbin ve kompresör blokları ayrı ayrı doğrulandıktan sonra bir sonraki aşama türbin ve kompresörü aynı şafta bağlayıp bir gaz jeneratörü modeli çözmektir. Böylece türbin ve kompresör eşleşmesi probleminin yazılım tarafından çözülüp çözülmediğinin testi yapılabilir. Şekil 37'de bu probleme ilişkin blok diyagramı gösterilmektedir.



Şekil 37. Türbin Kompresör Eşleşmesi Modeli Blok Diyagramı

Probleme ilişkin girdiler aşağıda belirtilmektedir.

- Rakım = 0 km
- Standart Gün Sıcaklık Farkı = 0 K,

Şekil 37'de yer alan "atmosfer bloğu" ISA Standart Atmosfer Modelini temsil etmektedir.

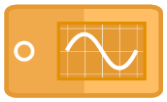
Tablo 9. Türbin Kompresör Eşleşme Modeli Sonuçlarının Karşılaştırması

	FlowNetMaster	TMATS
Kompresör Düzeltilmiş Kütle Debisi, lbm/s	242,24	220,46
Türbin Düzeltilmiş Kütle Debisi, lbm/s	248,86	227,1
Kompresör Düzeltilmiş Devir, devir/dakika	9.999	10.000
Türbin Düzeltilmiş Devir, devir/dakika	3.971	4.105
Kompresör Basınç Oranı	20,1	20,0
Türbin Basınç Oranı	3,0	3,0
Kompresör Verimi	0,850	0,842
Türbin Verimi	0,908	0,915
Kompresör Girişi Toplam Basınç, psi	14,4	14,4
Kompresör Çıkışı Toplam Basınç, psi	289,9	273,6
Türbin Çıkışı Toplam Basınç, psi	89,82	89,82
Kompresör Girişi Toplam Sıcaklık, R	518,7	518,7
Kompresör Çıkışı Toplam Sıcaklık, R	1233	1316
Türbin Girişi Toplam Sıcaklık, R	3193	3078
Türbin Çıkışı Toplam Sıcaklık, R	2501	2453
Kompresör Girişi Toplam Entalpi, BTU/lbm	124	124
Kompresör Çıkışı Toplam Entalpi, BTU/lbm	300	321
Türbin Girişi Toplam Entalpi, BTU/lbm	848	848
Türbin Çıkışı Toplam Entalpi, BTU/lbm	646	656

Sonuçlar Tablo 9'da karşılaştırılmaktadır, FlowNetMaster ile TMATS sonuçlarının oldukça yakın olduğu gözlenmektedir. Aradaki en çok %2-3'lük farkların ise çözüm toleransından kaynaklı olduğu düşünülmektedir.

SONUÇLARIN GÖRSELLEŞTİRİLMESİ

Zamana bağlı simülasyon sonuçlarının kullanıcılar tarafından program içerisinde hızlıca görselleştirilebilmesini sağlamak adına bir sanal osiloskop modeli ve spektrum analizörü oluşturulmuştur (Şekil 38).

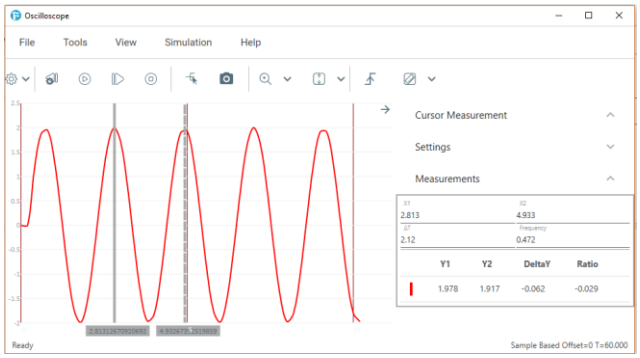


a. Osiloskop

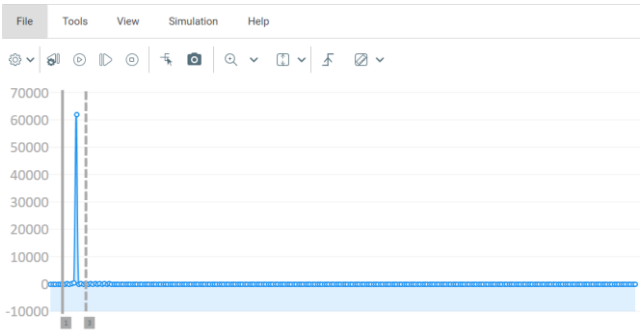
b. Spektrum Analizörü

Şekil 38. Osiloskop ve Spektrum Analizörü İkonları

Şekil 39'da osiloskop ekranının bir görüntüsü verilmektedir. Sinyaller model oluşturma alanında bu osiloskop ikonunun giriş portuna bağlanarak görüntülenebilmektedir. Benzer biçimde Şekil 40'ta ise spektum analizöründen bir ekran görüntüsü sunulmaktadır.



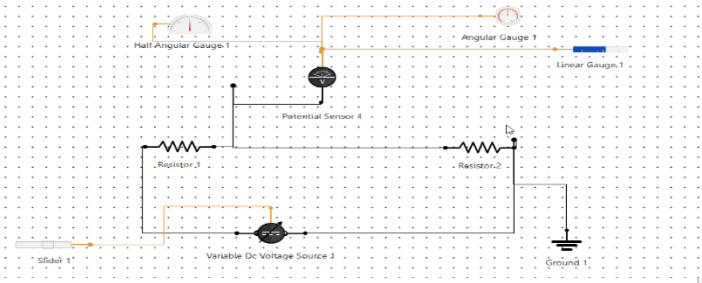
Şekil 39. Osiloskop Ekran Görüntüsü



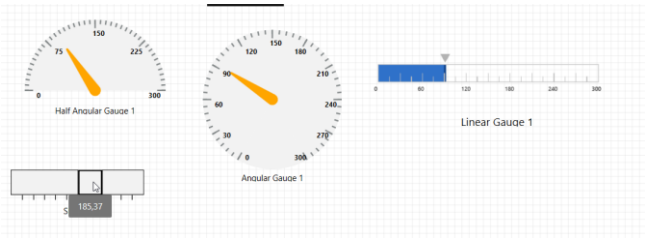
Şekil 40. Spektrum Analizörü Ektan Görüntüsü

KULLANICI İLE BENZETİM ARASINDAKİ ETKİLEŞİM

Kullanıcı ile benzetim arasındaki etkileşim "gösterge paneli" özelliği sayesinde sağlanmaktadır.Şekil 41'de örnek bir modele ait bir blok diyagram gösterilmiştir, Şekil 42'de ise bu modelin gösterge panelinin bir ekran görüntüsü sunulmaktadır. Bu panelde kullanıcı simülasyon değerlerini takip edebildiği gibi kontroller vasıtasıyla girdileri simülasyon esnasında değiştirebilmektedir.



Şekil 41. Gösterge Paneli Elemanlarını İçeren Örnek Bir Modelin Blok Diyagramı

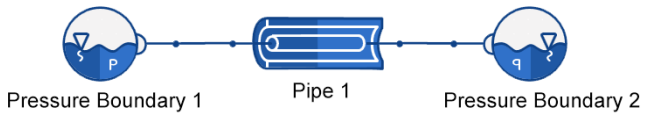


Şekil 42. Bir Önceki Şekilde Blok Diyagramı Verilen Modele Ait Gösterge Paneli Ekran Görüntüsü

UYGULAMA PROGRAMLAMA ARAYÜZÜ

Modeller grafik arayüzün dışında bir uygulama programlama arayüzü (API) vasıtası ile de oluşturulabilmektedir. API hem C# hem de Python programlama dilleri ile kullanılabilmektedir. Bunu yapabilmek için kodun çekirdek katmanını barındıran ".dll" dinamik kütüphane dosyasını Python ile ilişkilendirdir. Python ile .NET arasındaki bağlantı Pythonnet vasıtası ile sağlanır. API kodun çekirdek katmanının kendisi olduğundan bu katmanın dışarısı ile güvenli bir şekilde etkileşime girmesi için uygun bir biçimde enkapsülasyonu yapılmıştır.

API kullanılarak modellerin otomatik oluşturulması, betikler vasıtası ile otomasyon v.b. görevlerin kullanıcı açısından kullanımı hedeflenmiştir. Ayrıca API'ın sözdiziminin de kolay ve son kullanıcı açısından anlaşılabilir olmasına özen gösterilmiştir.



Şekil 43. İki Basınç Sınırı Arasındaki Bir Boru

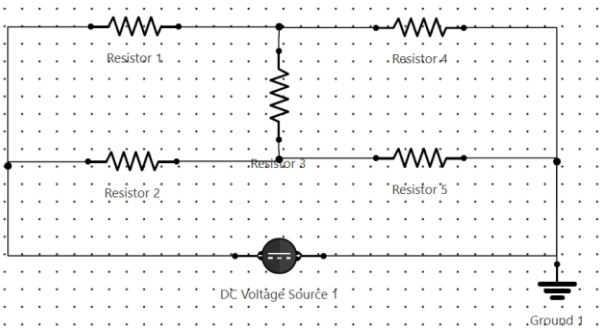
```python
import clr
import sys

clr.AddReference("FlowNetMaster.Core")
clr.AddReference("Units")

import FlowNetMaster.Core as fnm
import UnitsNet as unet
import UnitsNet.Units as units

model = fnm.Model()
fluidName = "WATER"

res_1_Temp=unet.Temperature(275.0,units.TemperatureUnit.Kelvin)
res_1_Pres=unet.Pressure(0.101,units.PressureUnit.Megapascal)
res_1_Medium= fnm.Utilities.Extentions.CreateMedium(res_1_Temp,
res_1_Pres,fluidName)
reservoir1=fnm.Node.Fluid.InfiniteReservoir(model,res_1_Medium)

res_2_Temp=unet.Temperature(275.0,units.TemperatureUnit.Kelvin)
res_2_Pres=unet.Pressure(0.105,units.PressureUnit.Megapascal)
res_2_Medium=fnm.Utilities.Extentions.CreateMedium(res_2_Temp,r
es_2_Pres,fluidName)
reservoir2=fnm.Node.Fluid.InfiniteReservoir(model,res_2_Medium)


pipe_Diameter=unet.Length(3.0,units.LengthUnit.Centimeter)
pipe_Length=unet.Length(16.0,units.LengthUnit.Meter)
pipe_Roughness=unet.Length(5.9e-02,units.LengthUnit.Inch)

pipe = fnm.Node.Fluid.Pipe(pipe_Diameter,pipe_Length,pipe_Rough
ness)

model.Add(reservoir1)
model.Add(reservoir2)
model.Add(pipe)

model.Connect(reservoir1.Port1, pipe.Port1)
model.Connect(reservoir2.Port1, pipe.Port2)

solver = fnm.Solver.SteadyState.NewtonRaphson()
result = model.Simulate(solver, bool(False))
```

Kod 1. İki Basınç Sınırı Arasındaki Boru Modeli için Python Kaynak Kodu

Şekil 43'de iki basınç sınırı arasındaki bir boru modeli gösterilmektedir. Bu modeli Python API ile oluşutup çözmek Kod 1'de verilen Python kaynak kod ile mümkündür. Kaynak kod incelendiğinde Units.Net (17) ve CoolProp (16) kütüphanelerinin programa nasıl entegre edildiğine ilişkin fikir sahibi de olunmaktadır.

Wheatstone Köprüsü

Şekil 44'te bilhassa ölçme sistemlerinde sıkça kullanılan bir elektrik devresi olan Wheatstone köprüsüne ait blok diyagramı gösterilmektedir. Kod 2'de ise bu modeli API kullanarak oluşturan Python kaynak kodu gösterilmektedir.



Şekil 44. Wheatstone Köprüsü Blok Diyagramı

FMI STANDARDI İLE EŞ-ZAMANLI BENZETİM

Functional Mock-up Interface (FMI) model tabanlı mühendislik simülasyonlarını standartlaştıran ve böylece eş zamanlı simülasyonları yapmayı sağlayan bir arayüzdür. Bu arayüz FMI konsorsiyumu tarafından tanımlanmıştır ve endüstri standardı haline gelmiştir. Bu standart sayesinde aynı zamanda entelektüel bilginin (fikri mülkiyet) açıklanmadan modellerin korunup paylaşılması mümkün olmaktadır.

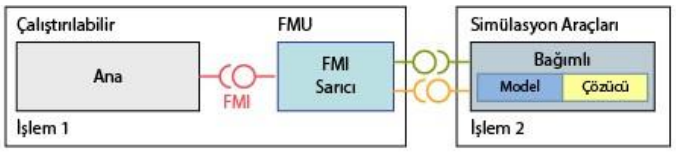
FMI arayüzü sayesinde kullanıcılar farklı uzmanlıkları olan ve FMI standartlarını sağlayan programları eş zamanlı olarak kullanarak geniş çaplı bir modeli simüle edebilme ikanı bulmaktadırlar. Böylece aynı modeli tekrar tekrar tasarlamak yerini uygulamalar arası veri aktarımı yapılarak zamandan büyük bir tasarruf edilmesi sağlanmaktadır. FMI standartları uygulanarak doğrudan model değişimi yapılabildiği gibi eş zamanla simülasyon da yapılabilmektedir. Model değişimi yapıldığında çözüm ana programda yapılmaktadır. Eş-zamanlı simülasyon yapıldığında ise çalıştırılan dosyaya ana yazılım kendi çözücüsünü de eklemektedir.

```
import clr
import sys

clr.AddReference("FlowNetMaster.Core")
clr.AddReference("Units")

import FlowNetMaster.Core as fnm
import UnitsNet as unet
import UnitsNet.Units as units


model = fnm.Model()

voltage = unet.ElectricPotential(100.0, units.ElectricPotential
Unit.Volt)
dcVoltageSource = fnm.Node.Electrical.DCVoltageSupply(voltage)

r1 = unet.ElectricResistance(100.0, units.ElectricResistanceUni
t.Ohm)
resistance1 = fnm.Node.Electrical.Resistance(r1)

r2 = unet.ElectricResistance(50.0, units.ElectricResistanceUnit
.Ohm)
resistance2 = fnm.Node.Electrical.Resistance(r2)
```

```
r3 = unet.ElectricResistance(100.0, units.ElectricResistanceUni
resistance3 = fnm.Node.Electrical.Resistance(r3)

r4 = unet.ElectricResistance(300.0, units.ElectricResistanceUni
t.Ohm)
resistance4 = fnm.Node.Electrical.Resistance(r4)

r5 = unet.ElectricResistance(150.0, units.ElectricResistanceUni
t.Ohm)
resistance5 = fnm.Node.Electrical.Resistance(r5)

ground = fnm.Node.Electrical.Ground()

model.Add(dcVoltageSource)
model.Add(resistance1)
model.Add(resistance2)
model.Add(resistance3)
model.Add(resistance4)
model.Add(resistance5)
model.Add(ground)

model.Connect(dcVoltageSource.Port1, resistance1.Port1, resista
nce2.Port1)
model.Connect(resistance4.Port1, resistance1.Port2, resistance3
.Port1)
model.Connect(resistance5.Port1, resistance2.Port2, resistance3
.Port2)
model.Connect(dcVoltageSource.Port2, ground.Port1, resistance4.
Port2, resistance5.Port2)


solver = fnm.Solver.SteadyState.NewtonRaphson()
result = model.Simulate(solver, bool(False))
```

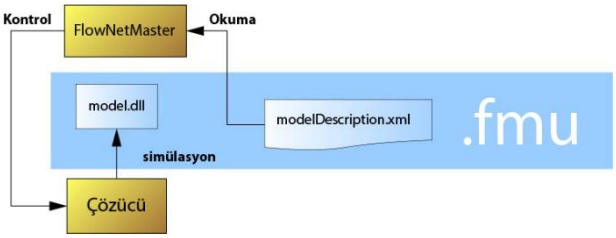
Kod 2. Wheatstone Köprüsü Python Kaynak Kodu

Eş-zamanlı benzetimin nasıl yapıldığı grafiksel olarak Şekil 45'te ele alınmaktadır. Geliştirilen yazılımda da şekilde tarif edilen yaklaşım uygulanmıştır.



Şekil 45. FMU ile Eş Zamanlı Benzetim

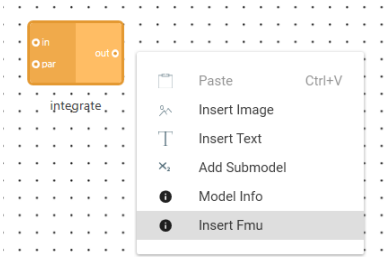
Functional Mock-up Unit (FMU) FMI standartlarını sağlayan bir dosya formatıdır. Yazılımlar arası veri aktarımı bu dosya formatı kullanılarak yapılmaktadır. Bu dosya sıkıştırılmış halde bulunur ve içerisine tanımlamaların bulunduğu bir "xml" dosyası ve simülasyonu çalıştıran "dll" uzantılı bir dosya gömülmüştür. FMU uzantılı dosyayı oluşturan yazılım "Ana Yazılım" konumunda bulunurken, bu dosyayı kullanan diğer yazılımlar "Bağımlı Yazılım" konumunda bulunmaktadır. Bu durum Şekil 46'da grafiksel olarak ifade edilmektedir.



Şekil 46. FMU Dosyasının İçeriği

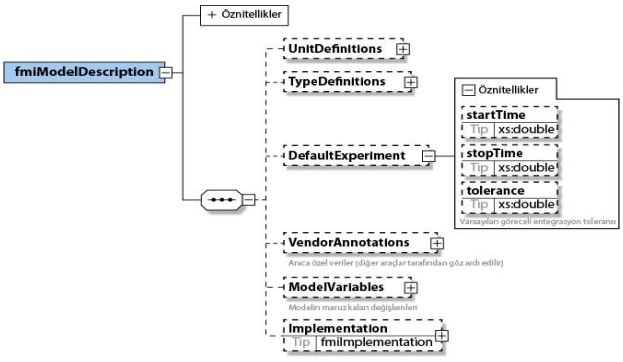
FlowNetMaster şu an "bağımlı yazılım" kipinde bu standartları desteklemektedir. Kullanıcılar FMI standartlarını uygulayan başka bir yazılımda oluşturdukları bir modeli fmu formatında kaydettikten sonra FlowNetMaster'a aktarıp, model içerisine bir alt model olarak rahatlıkla ekleyebilmektedirler.

Grafik arayüzde modelin blok diyagramına FMU bloğu eklenmesi Şekil 47'de gösterilmiştir.



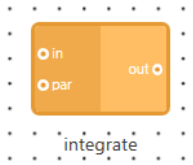
Şekil 47. Grafik Arayüzde Modele FMU Eklenmesi

Program kullanıcının seçtiği sıkıştırılmış fmu dosyasını açarak içerisindeki "modelDefinition.xml" dosyasını okur. Dosyanın içeriği ile ilgili ayrıntıları Şekil 48'de gösterilmektedir. Bu dosya çalıştırılacak model ile ilgileri bilgileri içerisinde barındırmaktadır. Okuma işlemi bitince bağlantı noktaları hakkındaki bilgiler Şekil 49'da görüldüğü gibi görsel üzerinde kullanıcıya gösterilmektedir. Burada kullanıcı giriş ve çıkış portlarını kolaylıkla görebilmektedir.



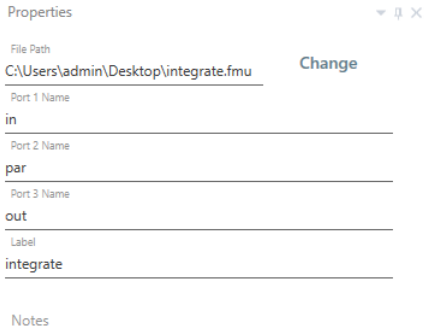
Şekil 48. FMI Model Tanımlaması

Şekil 49'da gösterilen ikon üzerinde yer alan "in" yazılı bağlantı noktaları çözülecek değerleri göstermektedir. "par" yazılı bağlantı noktaları ise sabit parametreleri göstermektedir. Son olarak "out" yazan bağlantı noktaları da çözüm sonucunda çıkan değerlerini göstermektedir.



Şekil 49. FMU Bloğuna Ait İkon

FMU dosyası programa eklendikten sonra özellik paneli üzerinden seçilen dosya ve port isimleri istenildiği gibi değiştirilebilir.



Şekil 50. FMU Özellik Paneli

FMU özellik paneli ise Şekil 50'de gösterilmektedir. Giriş ve çıkışların sayısı ve isimleri .fmu dosyası çözümlenirken otomatikman belirlenmektedir.

SONUÇ

Termal akışkan sistemlerinin model tabanlı tasarımı için nesne yönelimli yaklaşımlar kullanılarak katmanlı mimari yaklaşımla yeni bir yazılım geliştirilmiş ve açık literatürdeki çok sayıdaki doğrulama örnekleri ile verdiği sonuçlar doğrulanmıştır. Kullanıcının grafik arayüzde veyahut uygulama programlama arayüzü (API) ile tanımladığı model çekirdek katmanında bir diferansiyel cebirsel denklem takımına dönüştürülüp sayısal yöntemler ile çözülmektedir.

Yazılıma geniş bir akışkan kütüphanesi dahil edilmiş, kullanıcıların da bu kütüphaneye kendi tanımladıkları akışkanları dahil edebilmeleri sağlanmıştır. Yazılım standart komponentler için veritabanı desteği sunmaktadır. Örneğin pompa eğrileri kullanıcılar tarafından dijitize edilip veritabanına dahil edilebilmektedir.

Akış komponentlerini tarif eden bloklarının yanısıra ısı geçişi, kontrol sistemleri, mekanik, mantık, güç sistemleri, elektrik v.b. komponentler de modellenmiştir. Böylece akış sistemlerinin diğer sistemlerle etkileşimi de program tarafından modellenebilmekte ve kullacının istediği dijital ikiz oluşturulabilmektedir. Geliştirilen programın vana, pompa, kompresör ve daha birçok endüstriyel uygulamada model-tabanlı tasarım amacıyla faydalı olacağı düşünülmektedir.

TEŞEKKÜR

Çalışmaya maddi destek sağlayan Türkiye Bilimsel ve Teknik Araştırma Kurumu'na (TEYDEB Proje No: 2150113, 7180190, 1180195), ayrıca model tabanlı tasarım yaklaşımı üzerine yaptığımız son derece değerli tartışmalardan ötürü Doç.Dr. Çağlar Üçler'e, grafik arayüzün tasarımı konusunda Sn. Nagihan Aydınlık ve Sn. Alper Apaydın'a, testler konusunda verdiği destekten ötürü ise stajyerimiz. Mehmet Safa Gökalp'e teşekkürü borç biliriz.

KAYNAKLAR

1. Federal Highway Administration (FHWA), 2005, "Clarus Concept of Operations", Publication Sayı FHWA-JPO-05-072.
2. www.flownetmaster.com , Erişim Tarihi: 2019.
3. P. Paterno, 1999, "Model-Based Design and Evaluation of Interactive Applications", Springer-Verlag, Londra, s. 11.
4. S. Wang, K.G., Shin, 2006, "Task Construction for Model-Based Design of Embedded Control Software", IEEE Trans. Software Eng, Cilt 32, Sayı 4, sf. 254–265.
5. N.C. Horner, J.S., Topper, 2013, "Model-Based Systems Engineering in Support of Complex Systems Development", Johns Hopkins APL Technical Digest, Cilt 32, Sayı 1, sf. 419–432.
6. R. Plötzner, H.U. Hoppe, E. Fehse, C. Nolte, F. Tewissen, 2001, "Model-based Design of Activity Spaces for Collaborative Problem Solving and Learning", Proceedings of the European Conference on Artificial Intelligence in Education, Lizbon, Portekiz, sf. 372–378.
7. W.P M.H. Heemels, G. Muller, 2006, "Boderc: Model-based design of high-tech systems", Eindhoven Embedded Systems Institute, Hollanda.
8. N. Yang, G. Hua, J. Li-li, 2016, "Model-Based Design Methodology for Sampling Rate Converter", International Journal of Multimedia and Ubiquitous Engineering, Cilt 11, Sayı 5, sf. 83–92.
9. M.M.D. Santos, J.H. Neme, F.R. Franco, S.L. Stevan, S. L., W. Torres, A.B. Lugli, A.A.M. Lagana, J.F., Justo, 2015, "Model-Based Design of Exterior Lighting Control Function for Automobile – MIL", SIL and RCP, International Journal of Innovative Computing, Information and Control, Cilt 11, Sayı 5, sf. 1495–1507.
10. B. Kirby, L. Zou, J. Cao, I. Kamwa, A. Heniche, M. Dobrescu, 2011, "Development of a Predictive Out of Step Relay Using Model Based Design", Innovative Smart Grid Technologies – ISGT Europe Conference, IEEE, sf. 1–6.
11. S. Chatterjee, W.B. Kleijn, 2011, "Auditory Model-Based Design And Optimization Of Feature Vectors for Automatic Speech Recognition", IEEE Transactions on Audio, Speech, and Language Processing, Cilt. 19, Sayı. 6, sf. 1813–1825.
12. T. Miyajima, H. Fujimoto, M. Fujitsuna, 2013, "A Precise Model-Based Design Of Voltage Phase Controller for IPMSM", IEEE Transactions on Power Electronics, Cilt. 28, Sayı. 12, sf. 5655–5664.
13. M. Ahmadian, Z.J. Nazari, N. Nakhaee, Z. Kostic, 2005, "Model-Based Design and SDR", 2nd IEEE/EURASIP Conference, sf. 19–99.
14. S.S. Solano, M.B. Jimenez, E.D. Toro, P.B. Jimenez, I. Baturone, 2013, "Model-Based Design Methodology for Rapid Development of Fuzzy Controllers on FPGAs", IEEE Transactions on Industrial Informatics, Cilt 9, Sayı 3, sf. 1361–1370.
15. http://api.flownetmaster.com, Erişim Tarihi: 2019
16. http://coolprop.org Erişim Tarihi:2019
17. https://github.com/angularsen/UnitsNet Erişim Tarihi:2019
18. https://mathdotnet.com Erişim Tarihi 2019
19. B. E. Larock, R.W. Jeppson, G. Z. Watters, "Hydraulics of Pipeline Systems", 2000, CRC Press LLC
20. S. Kakaç, H. Liu, A. Pramuanjaroenkij, 2012, "Heat Exchangers: Selection, Rating and Thermal Design", Üçüncü Baskı, CRC Press, Boca Raton, Florida, ABD
21. J.W. Chapman, T.M. Lavelle, R.D. May, J.S. Litt, T.H. Guo, 2014, "Toolbox for the Modeling and Analysis of Thermodynamic Systems (T-MATS) User's Guide", NASA TM-2014-216638.

ABSTRACT

A new software, FlowNetMaster, for the creation of digital twins of thermal-hydraulic systems has been developed. FlowNetMaster has been developed using object-oriented programming paradigms in C# language within .NET Framework technology using MVVM (model-view-view model) architectural structure. Thanks to its database, standard components (pipes, pumps, fitting elements) can easily be selected by the user and new components defined by the user just as easily. In order to create the model, the user places components (represented by block icons) selected from the libraries (e.g. a heat exchanger, pipe etc.) onto the "model canvas" by drag and drop interactions and then connects the components to each other within this graphical environment. The software transforms this model into a set of differential algebraic equations in the core layer (model layer) and solve the behavior of the model in steady-state and transient regimes. More than one hundred built-in fluids (water, air, refrigirants, heat transfer liquids, anti-freeze solutions, heat transfer liquids) are defined inside the standard thermo-physical library of the software.

Furhermore, the user is also able to define a custom fluid. Thanks to the software's dashboard feature (human machine interface), users can interact with their models and monitor the indicators, change the value of the controls while their model is running.

The developed software (FlowNetMaster) can be used from the graphical user interface (GUI), as well as through an application programming interface (API) in order to create and solve models via Python or C#. End user documentation for the API is automatically generated and kept up-to-date.

The software has been developed in a modular fashion and divided into libraries and modules such as; fluid, signal, logic, heat transfer, mechanical, power, design optimization, application programming interface etc. These modules can be packaged separately or together.

Continous improvement and continuous deployment (CI/CD) strategies are utilized in the cloud to manage the software development process. Thus an integration that can satisfy end-user requests instantly has been established. This integration includes automated tests. Thus the integrity of the software and the accuracy of the results are continuously tested before the software reaches the user.

This paper briefly discusses the software development process and architecture, and presents several validation examples selected for various libraries. It is seen that the outcomes of the program are consistent with the data in the open literature. Therefore, it is concluded that the program can ber used to model and optimize industrial valve, compressor, pump systems and their controllers.