Mühendis ve Makina
cilt 60, sayı 695, s. 91-100, 2019
Derleme Makale

Engineer and Machinery
vol. 60, no. 695, p. 91-100, 2019
Review Article

Üretim Sistemleri Mühendisliğinde Kesintisiz Veri Alışverişi İçin Bir Çatı Formatı: AutomationML

Ender Yemenicioğlu[1]

ÖZ

Bu makale sektöründe önemli yer tutan firmalar tarafından geliştirilen ve mühendislik yazılımları arasındaki bilgi alışverişini sağlamayı hedefleyen bağımsız ve açık kaynaklı AutomationML veri formatının amaçlarını, uygulama alanlarını, temel kavramlarını ve mimarisini tanıtmak amacıyla yazılmıştır. AutomationML bir üretim sisteminin topolojik yapısını, bileşenlerinin geometrisini ve davranış tanımlamalarını saklamaya imkân sağlar. AutomationML, 2008 yılında Almanya çıkışlı olarak başladığı yolculuğuna onuncu yılının sonunda elliyi aşkın firma ve akademik kuruluşun işbirliğiyle dünya çapında bir konsorsiyum olarak devam etmektedir.

Anahtar Kelimeler: Üretim sistemleri, otomasyon, endüstri 4.0, automationml, mühendislik yazılımları, veri alışverişi

A Container Format for Seamless Data Exchange in Manufacturing Systems Engineering: AutomationML

ABSTRACT

This article introduces objectives, application areas, basic concepts and the architecture of the independent and open source AutomationML data format, which is developed by the important companies and institutions in the sector and aimed at providing data exchange between engineering software tools. AutomationML allows storing the topological structure of a production system, the geometry of its components, and behavioral descriptions. AutomationML started its journey in Germany in 2008 and is continuing as a worldwide consortium in cooperation with above fifty companies and academic institutions at the end of its tenth year.

Keywords: Manufacturing systems, automation, industry 4.0, automationML, engineering tools, data exchange

Geliş/Received : 04.01.2018
Kabul/Accepted : 22.11.2018

[1] Dr., -Ing., inpro Innovations. mbH ender.yemenicioglu@inpro.de

1. GİRİŞ

Üretim sistemleri dünyası önemli bir dönüm noktasındadır. Otomasyon teknolojilerindeki ilerlemenin hızlanması ve çevik (agile) imalat konsepti gereği müşteri ihtiyaç ve beklentilerinin planlamanın en başından sürece dâhil edilmesi, üretim sistemlerinde ürün portföyü ve kaynak kullanımıyla ilgili esnekliği artırma zorunluluğunu doğurmaktadır [1]. Fakat bu ulaşılması kolay bir hedef değildir. Endüstri 4.0 yaklaşımında öngörüldüğü gibi üretim sistemleri mühendisliğinde ve uygulamasında yeni yöntemler geliştirilmelidir [2, 3].

Endüstri 4.0 hedefi, üretim sistemlerinde dijital teknolojilerin yoğunlaştırılmasını ve hali hazırda yürütülen mühendislik faaliyetlerinin artan bir entegrasyonunu gerektiriyor. Fabrikanın planlama, işletmeye alma, üretim, söküm gibi farklı yaşam döngüsü evrelerinin, saha kontrolünden şirket ağı yönetimine kadar çeşitli kontrol katmanlarının ve mühendisler tarafından duruma uygun mühendislik (yazılım) araçları ile yürütülen mühendislik tasarımı aşamalarının birbiriyle olan uyumunu ve işbirliğini güçlendirmek bu noktada önem kazanıyor.

Artan esneklik ve dünya pazarıyla yaşanan rekabet, tasarımın daha sık değiştirilmesini ve hızlı yürütülmesini gerektirdiğinden mühendislik süreci için harcanan emek ve zaman diğer giderlere kıyasla giderek daha fazla yer tutmakta ve bir üretim sisteminin yaşam döngüsünde önemli bir maliyeti oluşturmaktadır. Mühendislik zamanını ve maliyetlerini düşürmenin bir yolu tasarım ve planlama faaliyetlerinin iş birliğini artırmaktan geçiyor. Bunun için de kullanılan (yazılım) araçlarının mühendislik süreci boyunca kesintisiz bir biçimde bilgi alışverişi yapabilmesini sağlamak gerekmektedir. Böylece mühendislik sürecinin verimliliği, mühendislik yazılımlarının birbiriyle uyumunun artması sonucu mühendislik süreçleri arasındaki geçişin kolaylaşması ve hatta eşzamanlı yürütmenin mümkün hale gelmesiyle artırılmış olur [4]. Tipik bir üretim sistemi planlama sürecinde her geçen gün giderek artan yazılım çeşitliliği ve her bir uygulamanın kendine has farklı veri formatlarını kullanması bu hedefi maalesef zorlaştırmakta, verilerin aktarımının önemli bir gider kalemi olmasına sebep olmaktadır [5, 6].

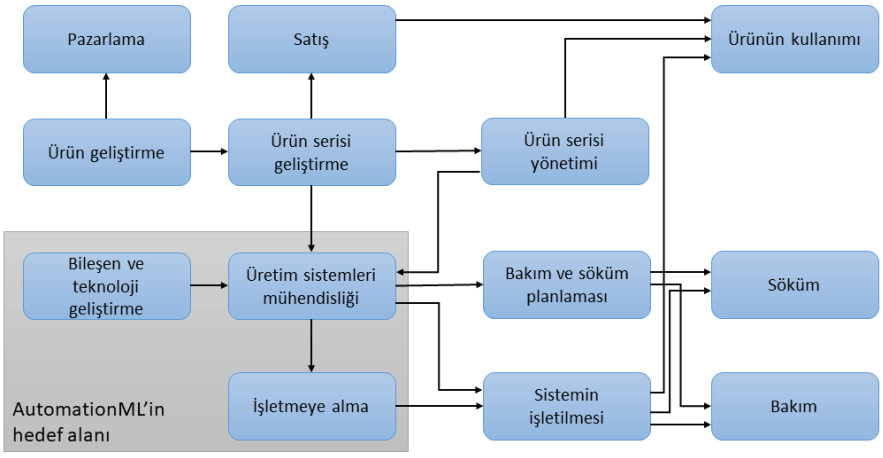
2008 yılında bahsedilen bu problemi çözmek için Daimler AG öncülüğünde ABB, Siemens, Rockwell, Kuka, netAllied, Zühlke firmaları ve Karlsruhe ve Magdeburg üniversiteleri mühendislik yazılımları için Automation Markup Language (AutomationML/AML) adı verilen kurumlardan ve şirketlerden bağımsız, açık kaynaklı bir veri alışverişi formatı oluşturmak üzere bir konsorsiyum kurmaya karar verdi [7]. Geçen süre içinde BMW Group, Volkswagen AG, Festo, Huawei, Airbus, Mitsubishi Electric, ThyssenKrupp gibi sektöründe önemli yer tutan pek çok firmanın ve başka akademik kuruluşların da katılımıyla konsorsiyumdaki üye kuruluş sayısı 2019 yılı itibariyle 55'i bulmuş ve Almanya sınırlarını aşmış bulunmaktadır.

Bu yazının amacı AutomationML'in temel kavramlarını kısaca tanıtmak ve örneklerle açıklamaktır.

2. AUTOMATIONML'İN TEMEL KAVRAMLARI

Üretim sistemlerinin mühendisliği, çok çeşitli disiplinlerden mühendislerin yer aldığı, farklı planlama ve tasarım süreçlerinin yürütüldüğü ve bir imalat sisteminin kurulması, çalıştırılması, muhafaza edilmesi ve en sonunda sökümü için gerekli olan mühendislik ürünlerinin kullanılmasını ve oluşturulmasını içeren karmaşık bir süreçtir [1, 9].

AutomationML'in ana hedefi, bu süreçte yer alan üretim elemanlarının yaşam döngüleri içinde üretilen verileri kapsamaktır. Burada öne çıkan aşamalar üretim sistemi bileşenlerinin edinilmesi veya tasarlanması, bunların ilgili imalat işlemlerinde kullanılması, sistemin fabrika içindeki kurulumunun detaylı olarak planlaması ve son olarak işletmeye alınması olarak sıralanabilir (bkz. Şekil 1) [4].



Şekil 1. AutomationML'in Kapsamına Giren Mühendislik Süreci [4]

AutomationML kendisini sınırladığı hedef alanında bile farklı mühendislik disiplinlerini, çalışma prensiplerini, iş ve adlandırma kurallarını kapsamak zorundadır. Bu geniş kapsama ve karmaşıklığa hakim olabilmek için güvenilir, iyi denenmiş bir yöntem ise nesne yönelimi (object orientation) yaklaşımıdır. AutomationML nesne yönelimine uygun olarak üretim sistemi elemanlarını veri nesneleri olarak tanımlar. Bir nesne onu tanımlayan niteliklere sahip olabilir, başka alt nesnelerden oluşabilir ve kendisi daha büyük bir bileşimin parçası olabilir. Bu tür veri nesnelerine örnek olarak robotlar, motor, kol, kıskaç gibi alt robot öğeleri, yağ hazneleri, programlanabilir mantıksal denetleyiciler, sinyaller, sinyal verileri hatta komple üretim istasyonları verilebilir.

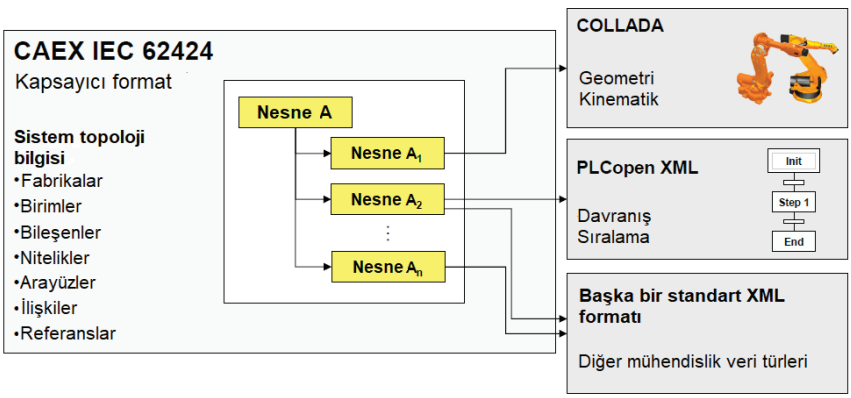
Veri nesnesiyle temsil edilen üretim elemanının saklanmaya en çok ihtiyaç duyulan özellikleri hiyerarşik üretim topolojisi içindeki konumu, geometrisi, kinematiği, mantık verileri (sıralama, davranış ve kontrol) ve diğer nesnelerle olan ilişkileridir [7].

Bu özellikleri temsil etmek için hali hazırda veri formatları zaten mevcut olduğundan AutomationML bunları temel alarak "olduğu gibi" kullanmaktadır, tekrardan bir tanımlamaya gitmemektedir. Yani AutomationML önceden mevcut olan ve kabul görmüş farklı veri standartları arasında işbirliği sağlamakta ve üretim sistemi bileşenlerini tanımlamak için bunların nasıl kullanılacağını tarif etmektedir [10]. Şekil 2'de AutomationML'in genel mimari yapısı ve temel bileşenleri gösterilmiştir.

AutomationML'in temelini kapsayıcı format olan XML tabanlı CAEX [11] oluşturmaktadır. Üretim sistemlerinin modellemesi için CAEX üzerinden yine XML tabanlı COLLADA [12] ve PLCopen XML [13] formatlarına referanslar verilmekte ve AutomationML çatısında bütünleştirilmektedir. Gerek duyulduğunda başka bir standart XML dokümanı da bu yapıya eklenebilir.

AutomationML aşağıda sıralanan temel kavramların depolanmasını desteklemektedir [4, 7, 10]:

Sistem topolojisi: Sistem topolojisi, üretim sistemlerini alt bileşenlerinin hiyerarşik bir yapısı olarak tanımlar. CAEX formatı üzerinden tanımlanan bu yapı, AutomationML'in kapsayıcı öğesidir. Bir üretim elemanının özellikleri, doğrudan ilgili AutomationML veri nesnesiyle ilişkilendirilir; örneğin bir robot tutucusunun konumu, geometrisi, davranış bilgisi ve benzeri. Her nesnenin kendine has özellikleri ve



Şekil 2. Mühendislik Veri Alışverişi Formatı AutomationML'in Genel Yapısı [10]

arayüzleri vardır. AutomationML, CAEX'in üretim sistemleri için kullanımını ayrıntılı olarak tanımlar. Bu konu bir sonraki bölümde detaylı olarak açıklanacaktır.

Geometri ve kinematik: AutomationML bir nesnenin geometri ve kinematik bilgilerini saklamak için COLLADA 1.4.1 ve 1.5.0 uluslararası standartlarını kullanmaktadır. Bu veri tipinin modellenmesi iki aşamalı olarak gerçekleşir. Önce ilgili nesnenin geometrisi ve kinematik davranışı COLLADA formatında tanımlanır. Sonra, bu dosyalara ve/veya içindeki veri nesnelerine CAEX belgesinde referans gösterilir.

Kontrol bağlantılı mantık verileri: Nesnelerin davranışı bir eylem sırası olarak tanımlanabilir (ör. başlat→yürüt→dur). Bu veri tipi, sinyal giriş çıkış tanımlamalarını ve mantıksal değişkenleri içeren servis fonksiyon zincirleri (service function chain, SFC) olarak PLCopen XML [3] veri formatı vasıtasıyla saklanır. Bu fonksiyon zincirlerine bağlı değişkenlere CAEX içinden referans verilebilir, böylece CAEX ve PLCopen XML arasında bir arabağlantı oluşturulur. İmalat işlemlerini tanımlamak için makina mühendisleri tarafından sıkça kullanılan akış şemaları (Gantt çizelgeleri) bile PLCopen XML içinde basitleştirilmiş şekilde servis fonksiyon zinciri olarak saklanabilir ve daha sonraki bir mühendislik evresinde programlanabilir mantıksal denetleyici (PLC) koduna dönüştürülebilir.

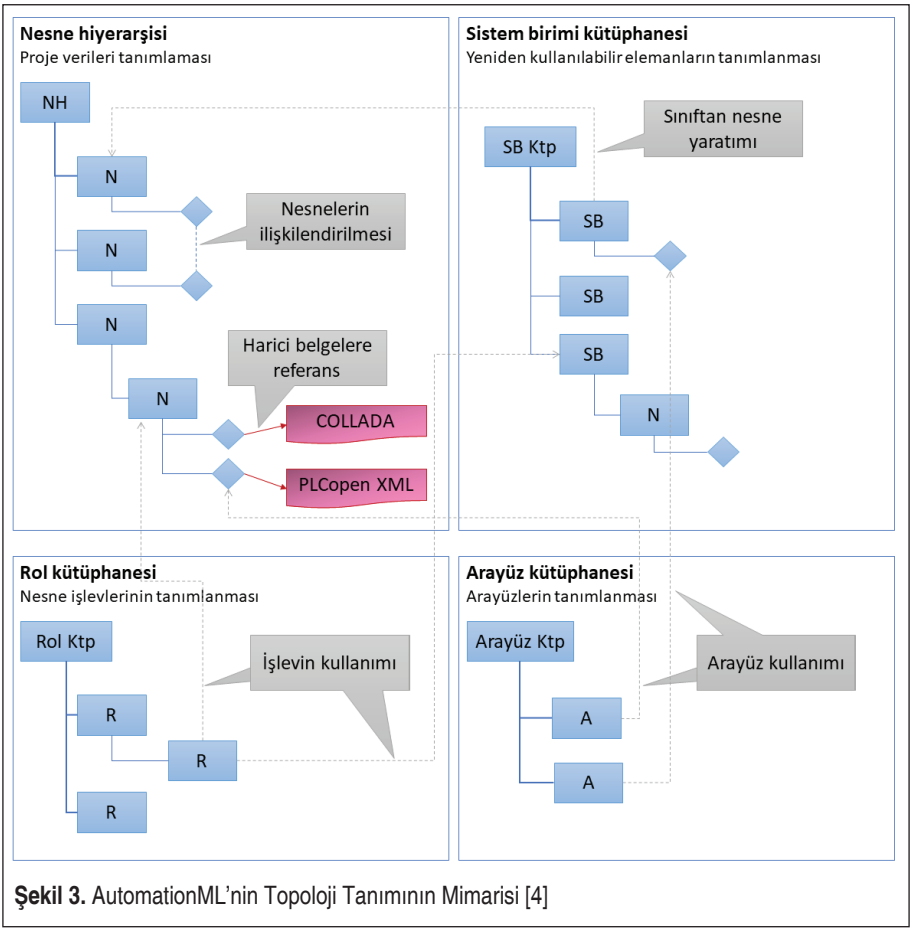
Referanslar ve ilişkiler: AutomationML, nesneler arasındaki ilişkiler ve referanslar arasında ayrım yapar. Referanslar CAEX nesnelerinden harici olarak depolanmış dosyalara tanımlanır (ör. bir robot nesnesinin geometrisi için harici bir dosyaya verilen referans). Öte yandan ilişkiler CAEX nesneleri arasındaki karşılıklı ilişkileri örneğin bir giriş/çıkış kartının belirli bir kanalıyla bir robot sinyali arasındaki bağlantıyı belirtmektedir. Nesneler arasındaki ilişkiyi oluşturmak için, iki nesneyi birbiriyle ilişkilendiren arayüzün de CAEX içinde tanımlanması gerekir. AutomationML üretim sistemlerinde sık kullanılan bazı standart arayüzleri içermektedir.

3. AUTOMATIONML'DE BULUNAN VERİ TÜRLERİ

AutomationML'nin temel yapısının önemli bir kısmı CAEX'in sınırlandırılmış bir uygulamasıdır. CAEX farklı mühendislik alanlarından gelen veri türlerini depolamak için esnek mekanizmalar sağlarken, AutomationML bunların üretim mühendisliği için somut uygulamasını tanımlar. AutomationML aşağıdaki özel nitelikleri, sınıfları, arabirimleri ve nesne tanımlama kurallarını içermektedir [4, 7, 10]:

Nesne/Sürüm tanımlama: Nesneler genel benzersiz kimlik (global unique identification) aracılığıyla tanımlanır. Ayrıca dokümanlar için sürüm numarası ve sürüm geçmişi bilgileri de dahil olmak üzere sürüm bilgisinin saklanması için mekanizmalar sağlanmıştır.

Rol kütüphanesi (RoleClassLib): Bir rol sınıfı, bir üretim sistemi bileşeninin genel işlevini tanımlar. Örnek olarak robot, konveyör, mantıksal denetleyici gibi genel fonk-



Şekil 3. AutomationML'nin Topoloji Tanımının Mimarisi [4]

siyonel tanımlar verilebilir. Rol sınıfları ilgili bir rol kütüphanesinin içinde bulunur.

Sistem birimi kütüphanesi (SystemUnitClassLib): Kullanıcıların tanımladığı AutomationML nesne sınıfları sistem birimi kütüphaneleri içinde bulunur. AutomationML standart olarak belirli bir sistem birimi kütüphanesi içermez, ancak sistem birimi sınıflarını modellemek için gereken kuralları tanımlar. Yazım ve içerik kurallarının yanı sıra başka AutomationML veri türlerine yapılan referanslar ve sınıflardan nesne yaratmanın nasıl yapılacağı da bunlara dâhildir.

Arayüz kütüphanesi (InterfaceClassLib): Arayüzler, nesneler arasındaki ilişkilerin açıklaması için kullanılır. AutomationML üretim/otomasyon sistemleri modellemesinde kullanılmak için bir dizi genel arayüz sınıfı içeren bir arayüz kütüphanesi sunar. Arayüz sınıflarından yaratılan arayüz nesneleri diğer veri nesnelerinin bir alt öğesi olarak tanımlanabilir.

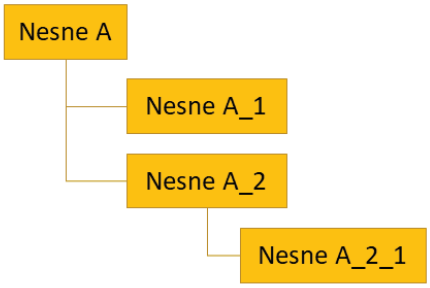
Nesne Hiyerarşisi (InstanceHierarchy): Nesne hiyerarşileri somut proje verilerini depolar ve bu nedenle AutomationML'nin en önemli kısmıdır. Nesne hiyerarşisi sistem elemanlarını topolojik konumu, özellikleri, arayüzleri, referansları ve ilişkileri ile tanımlayarak temel gövdeyi oluşturur.

4. UYGULAMA ÖRNEKLERİ

Basit bir nesne hiyerarşisinin AutomationML'de nasıl tanımlandığı Şekil 4'te gösterilmektedir. Bir hiyerarşi iç içe geçmiş dâhili elemanlar (ing. InternalElement) aracılığıyla oluşturulur. Her dâhili eleman keyfî verilen bir ad ve benzersiz bir kimlik (guid) ile karakterize edilir [10].

Sistem nesneleri arasındaki ilişkilerin nasıl tanımlandığıyla ilgili olan bir örnek ise Şekil 5'te göstermektedir. Hiyerarşide «Robot» ve «Mantık denetleyici» adında iki nesne vardır. Her iki nesne de harici arayüz (externalInterface) elemanıyla tanımlanan bir sinyal arabirimine sahiptir. Dahili bağlantı (internalLink) adı verilen AutomationML elemanı ile bu iki arayüz arasında bir bağlantı tanımlanır.

Bir AutomationML nesnesi geometrik veya kinematik olarak modellenecekse, bu veriler ayrı bir COLLADA dosyasında bulunmalıdır. Daha sonra ilgili dâhili eleman standart AutomationML arayüzü "COLLADAInterface" üzerinden bu dosyaya referans verir. Buna ek olarak, bir nesnenin konum bilgisi üç boyutlu düzlemde öteleme ve dönme miktarı verilerek tanımlanır. Bunun için de çerçeve (frame) adındaki



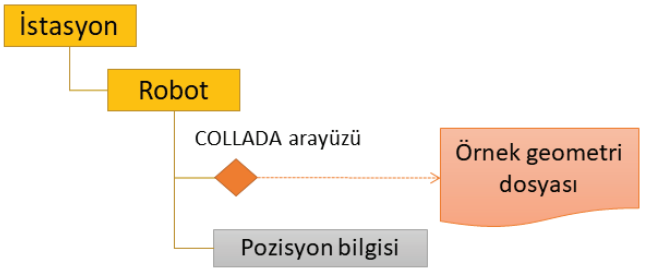

```
<InstanceHierarchy Name="Üst alt ilişkileri örneği">
    <Version>1.0.0</Version>
    <InternalElement Name="Nesne A" ID="360bb384-b853-4178-a1b3-5f310271fd9a">
        <InternalElement Name="Nesne A_1" ID="7b7d6a90-189a-42dc-a3ce-40e0ffdaca6f" />
        <InternalElement Name="Nesne A_2" ID="ec23623a-847a-4bee-bb1c-ebb84fff057b">
            <InternalElement Name="Nesne A_2_1" ID="8525d933-c306-4b71-b98e-9db49c120089" />
        </InternalElement>
    </InternalElement>
</InstanceHierarchy>
```

Şekil 4. Nesne Hiyerarşisi Örneği


```
<InstanceHierarchy Name="Bağlantı örneği">
    <Version>1.0.0</Version>
    <InternalElement Name="İstasyon" ID="360bb384-b853-4178-a1b3-5f310271fd9a">
        <InternalElement Name="Robot" ID="7b7d6a90-189a-42dc-a3ce-40e0ffdaca6f">
            <ExternalInterface Name="Baslat" ID="05794d07-021e-422d-ab72-9dcad46df836" />
        </InternalElement>
        <InternalElement Name="Mantıksal denetleyici" ID="ec23623a-847a-4bee-bb1c-ebb84fff057b">
            <InternalElement Name="Kart 1" ID="8525d933-c306-4b71-b98e-9db49c120089">
                <ExternalInterface Name="Kanal 1" ID="f8f5a3bb-1c39-42a5-8cff-d479bd9c0771" />
            </InternalElement>
        </InternalElement>
        <InternalLink Name="Bağlantı 1" RefPartnerSideA="7b7d6a90-189a-42dc-a3ce-40e0ffdaca6f:Baslat"
            RefPartnerSideB="8525d933-c306-4b71-b98e-9db49c120089:Kanal 1" />
    </InternalElement>
</InstanceHierarchy>
```

Şekil 5. AutomationML Nesneleri Arasında Bağlantı Örneği

standart AutomationML nesne niteliği kullanılır. Geometri referansının uygulaması Şekil 6'da gösterilmiştir.




```
<InstanceHierarchy Name="Geometri referansı örneği">
    <Version>1.0.0</Version>
    <InternalElement Name="İstasyon" ID="360bb384-b853-4178-a1b3-5f310271fd9a">
        <InternalElement Name="Robot" ID="7b7d6a90-189a-42dc-a3ce-40e0ffdaca6f">
            <Attribute AttributeDataType="" Name="Frame">
                <Attribute Name="x" AttributeDataType="xs:double">
                    <Value>1</Value>
                </Attribute>
                <Attribute Name="y" AttributeDataType="xs:double">
                <Attribute Name="z" AttributeDataType="xs:double">
                <Attribute Name="rx" AttributeDataType="xs:double">
                <Attribute Name="ry" AttributeDataType="xs:double">
                <Attribute Name="rz" AttributeDataType="xs:double">
            </Attribute>
            <ExternalInterface Name="COLLADA referans arayüzü" RefBaseClassPath="COLLADAInterface"
                ID="0b77e750-4c6a-40f9-85b3-4bc6c8aa27e1">
                    <Attribute Name="refURI" AttributeDataType="xs:anyURI">
                        <Value>.../OrnekGeometri.dae</Value>
                    </Attribute>
            </ExternalInterface>
        </InternalElement>
    </InternalElement>
</InstanceHierarchy>
```

Şekil 6 Harici Bir Geometri Dosyasına Referans Verme

AutomationML nesnelerinin davranış modelleriyle ilişkilendirilmesi gerekirse geometri dosyasına verilen referansa paralel bir şekilde mantık modelini içeren PLCopen XML dosyasına referans verilmesi gerekir. Bu iş için ise "PLCopenXMLInterface" arayüzü kullanılır.

5. SONUÇ

AutomationML, üretim/otomasyon mühendisliğinde kesintisiz veri aktarımını sağlamak amacıyla oluşturulmuş XML tabanlı bir veri formatıdır. Bu tür bir veri formatına duyulan ihtiyaç, mevcut mühendislik yazılım araçlarının veri aktarımındaki sınırlı kapasitesinden ve birbiriyle olan uyumsuzluğundan kaynaklanmıştır.

Bir üretim sisteminin en baştan itibaren nasıl tasarlandığı düşünüldüğünde AutomationML'nin nasıl bir boşluğu doldurduğu da net olarak görülebilir. Bir mühendislik projesinin başlangıcında, gerekli mekanik bileşenlerin üç boyutlu geometrisi ve kinematiği bir üç boyutlu tasarım programı kullanılarak oluşturulur. AutomationML aracılığıyla, bu geometri bilgileri bir üretim simülasyonu programına aktarılabilir. Simülasyon yardımıyla üretim birimlerinin konumları, malzeme akışı ve lojistik optimize edilerek planlanır. Buna paralel olarak, makina mühendisleri üretim için gerekli işlem dizilerini hazırlar. Daha sonra elektrik/kablo planlaması yapılır, mantıksal denetleyici ve robot programları oluşturulur. Bu bilgilerin hepsi AutomationML tarafından saklanabilir ve bir sonraki planlama aşamasına taşınır. Son olarak geometri, kinematik, işlem sırası ve giriş çıkış sinyalleri verileri içeren bir üretim istasyonu komple sanal devreye alma (virtual commissioning) ortamına taşınıp henüz fiziki olarak kurulmadan test edilebilir [7]. Ayrıca, bu veriler sanal ortamda personel eğitimi, standardizasyon faaliyetleri için, cihaz bakımlarının desteklenmesi veya istasyonda yapılan değişikliklerin önceki haliyle karşılaştırılması yeniden kullanılabilir.

Bu makalede mühendislik yazılımları arasındaki veri aktarımı sorununu optimum bir şekilde çözüme kavuşturmak için 2008 yılından bu yana geliştirilen AutomationML formatı kısaca tanıtılmıştır. Temel mimarinin, veri türlerinin ve kütüphane elemanlarının anlatımının yanı sıra farklı uygulama örnekleri de verilmiştir. AutomationML kullanımı Almanya'da özellikle araba sanayisi içinde giderek yaygınlaşmaktadır. Sanayideki uygulama örnekleri daha detaylı bir yazının konusu olarak incelenebilir.

KAYNAKÇA

1. **W. Terkaj, T. Tolio, A. Valente.** 2009. "Focused Flexibility in Production Systems," Changeable and Reconfigurable Manufacturing Systems, p. 47-66.

2. **H. Kagermann, J. Helbig, A. Hellinger, W. Wahlster.** 2013. "Umsetzungsempfehlungen für das Zukunftsprojekt Industrie 4.0: Deutschlands Zukunft als Produktionsstandort sichern," Abschlussbericht des Arbeitskreises Industrie 4.0, Forschungsunion.

3. **J. Jasperneite.** 2012. "Industrie 4.0–Alter Wein in neuen Schläuchen" Computer&Automation, Ausgabe: 2012/12, S. 24-28.

4. **A. Lüder, N. Schmidt.** 2017. "AutomationML in a Nutshell," Handbuch Industrie 4.0, no: 2, Berlin, Springer, p. 213-258.

5. **A. A. Garcia, R. Drath.** 2007. "AutomationML™ verbindet Werkzeuge der Fertigungsplanung–Hintergründe und Ziele," ABB Forschungszentrum, Ladenburg.

6. **J. Kiefer.** 2007. "Mechatronikorientierte Planung automatisierter Fertigungszellen im Bereich Karosserierohbau," Universität des Saarlandes, Saarbrücken.

7. **R. Drath, A. Lüder, J. Peschke, L. Hundt.** 2008. "AutomationML-the Glue for Seamless Automation Engineering," Emerging Technologies and Factory Automation ETFA 2008, IEEE International Conference.

8. AutomationML Consortium. "AutomationML Members" AutomationML e.V., www.automationml.org/o.red.c/mitglieder.html, son erişim tarihi: 2 Ocak 2018.

9. **T. Schaeffler, M. Foehr, R. Kodes ve A. Lüder.** 2014. "Regionalization of Engineering," 20th ICE Conference Proceedings, Bergamo, Italy.

10. AutomationML Consortium. 2016. "Whitepaper AutomationML Part 1 - Architecture and General Requirements" www.automationml.org/o.red/uploads/dateien/1485867599-AML_Whitepaper_Architecture_V2.0.0.zip, son erişim tarihi: 2 Ocak 2018.

11. International Electrotechnical Commission. "IEC 62424 - Representation of Process Control Engineering - Requests in P&I Diagrams and Data Exchange Between P&ID Tools and PCE-CAE Tools," 2016.

12. The Khronos Group Inc. 2008. "COLLADA – Digital Asset Schema Release 1.5.0 Specification," www.khronos.org/files/collada_spec_1_5.pdf, son erişim tarihi: 3 Ocak 2018.

13. PLCopen. "PLCopen XML," www.plcopen.org/pages/tc6_xml/xml_intro, son erişim tarihi: 3 Ocak 2018.