**Çevrim içi modelleme dili kullanıcı dökümanı**

Web tabanlı model geliştirme ortamı modelden modele ve koda dönüşümü hedefler. İlgili [bağlantı](https://web-dsml.vercel.app) üzerinden ulaşıldığında uygulama ana ekranına herhangi bir yetkileme olmadan girişi sağlanır.

Uygulamanın sol panelinde birden fazla projenin oluşturulabileceği bir “projects” dizini bulunur. Sağ taraftaki panelde ise modellemenin yapılabilmesini sağlayan “node” ve “edge” bileşenlerinin bulunduğu bir alan vardır. Yine aynı panelin devamında ilgili dizaynın json model’e dönüşümünü sağlayacak olan bir button ve seçili “node” isminin değiştirilebileceği bir alan mevcuttur.



**Node tipleri**

**Agent Node**; Bir mas modeli oluşturulması sırasında ajan tanımının yapılması için kullanılır.

**Capability Node;** Bir mas modeli içinde tanımlanan ajanların yeteneğinin tanımlanmasını sağlar.

**Environment Node;** Bir mas modeli içerisinde AOSE ajanlarının ortamlarının tanımlanması için kullanılır.

**Event Node**; Ajanların ortamlar ile gireceği etkileşim anlarının betimlenmesinde kullanılır.

**Operation Node**; Her bir olay için gerçekleştirilecek operasyonların tanımlanması sağlanır.

**Belief Node**; Ajanların yeteneklerine bağlı olarak yapabileceği düşünülen inançların tanımıdır.

**Plan Node**; Ajan yeteneklerine bağlı olarak planlanan davranışların oluşturulması sağlanır.

**Relation Node;** Mas tasarımı içerisinde her bir düğümün birbiriyle olan ilişkisini tanımlar. Bir üst modelde var olan ilişki tanımlarını bir alt modelde tanımlama yapılabilmesini sağlayan joker bir node gibi tanımlanabilir. Örneğin; var olan bir “capability” modeli içerisinde bir üst “environment” model ilişkisi kurulmak istendiğinde referans dosya ismi : node Adı (GarbageWorld:TrashOccured) şeklinde bir tanım yapılabilir.

**Action Node**; Ajan yetenek ve olay sonuclarına gore alınabilecek aksiyonların tanımını içerir.

**Message Node;** Ajan yetenek ve olay sonuclarına gore alınabilecek sonucları içerir. Bir çeşit aksiyon düğümü olmasına rağmen çıktı vermek üzerine bir görev üstlenir.

Edge Tipleri;

**InteractsWith**; Ajanların bulundukları olrtamla etkileşimlerini tanımlamak üzere oluşturulmuştur.

**Implements**; Ajanların yeteneklerinin tanımlarını ve ilişkilerini belirtmek için kullanılır.

**SubCapability**; Bir ana yetenek tanımının alt yeteneklerinin bulunabileceği zamanlarda alt yetenek tanımını ilişkilendirmek üzere çalışır.

**PostEvents**; Olay tanımlarının operasyon sonucları ile ilişkisi için kullanılır.

**HasContext**; İnanç ve Plan tanımlarının ilişkilendirmesini sağlar.

**IsTriggered**; Bir plan dahilinde hedeflenen olayın oluşmasını sağlamak ve bunu tanımlamak için kullanılabilir.

**Use**; İnanç ilişkisi yaratmak için kullanılır.

**AddBelief**; İnanç ilişkisi oluşturur.

**DelBelief**; Var olan İnanç ilişkisini gerçekleşen olay sonucunda silmesi için kullanılır.

 **Örnek bir Çöp toplama uygulamasının tasarımı**

“Projects” dizini altında oluşturulabilecek projenin ismi (GarbageCollector) ile bir alt dizin oluşturulur. Çöp toplama uygulamasının tasarımını sağlayacak ilgili tasarım bileşenleri SAML uygulama dili temel alınarak geliştirilebilir.

Bu bileşenler; Mas.**mas**, Environment.**env**,Capability.**cap**,Plan.**pln**



GarbageCollector için örnek bir proje ağacı gösterimi

Her bileşene ait tasarım sayfası proje ağacındaki konumuna göre tek tıklama ile açılabilir. Bu bileşenler temel alınarak, ilgili node ve edge alanları dizayn panelinde otomatik olarak gösterilir. Burada önemli olan nokta ise her tasarım bileşeninin mutlaka doğru dosya uzantıları ile oluşturulmasıdır. (mas.**mas**,environment.**env** gibi)Doğru dosya adı ile oluşturmadığınızda uygulama sizi uyaracaktır.



Garbage world MAS tasarım alanından bir görünüm ve node isim güncellemesi

Temelde yeni bir tasarım oluşturabilmek için ilgili projenin alt dizininde oluşturduğunuz dosya içerisinde, ilgili “nodes” bileşenleri sürükle bırak yöntemi ile eklenebilir iken bağlantılı olabilecek “edges” bileşenleri ile tek tıklama ile tanımlanabilir. Bu bileşenlerin üzerinde fare imleci tutulduğunda isimlendirilmesi anlaşılabilecektir. Model tasarım alanında oluşturulan her node ve edge bileşeni isim ve icon ile takibi sağlanabilir.

İlgili tasarım ekranında süreç tamamlandıktan sonra sağ üst köşede “**save**” buttonu veya “**Ctrl+S**” kısa yolu ile ile kaydedilerek ilgili progress icon’u ile kaydedildiğinden emin olunmalıdır. Tasarım alanından silinmek istenen node veya edge bileşeni; üstüne tıklandıktan sonra “**backspace”** tuşu ile silinebilir.

Her oluşturulan node, varsayılan bir node ismi ile birlikte oluşturulur. Bu isimlerinin düzenlenebilmesi için ilgili node üzerine tıklanarak “Update Nodes” alanından düzenleme yapılması gerekir.

İlerleyen safhalarda “Mas” modeli içerisinde oluşturulan bileşen isimleri ile bu bileşenlere ait dosya isimlerinin aynı olması gerektiğine dikkat ediniz. Örnegin(**Cleaning**.cap, **GarbageWorld**.env,**Moving**.cap)



Garbage world environment bileşeni tasarım ekranından bir görünüm.

Garbage world uygulaması gerekliliği olarak bir environment(Garbageworld.env) dosyası oluşturularak bu dosyanın tasarım ekranında sürükle bırak yöntemi ile ilgili node bileşenleri eklenmiştir. Node bileşenleri için ilgili edge bağlamları designer toolbox alanından tek tık ile seçilerek nodelar arasında bağlantı yapılmıştır.



 MAS metamodelinde bulunan “Cleaning” capability bileşenine ait detay tasarım ekranı.

Garbage collector için tanımlanan bir temizleme yeteneğine ait inanç, plan gibi node bileşenleri bulunur.Mas metamodelinde oluşturulan node ismi ile capability dosya isminin aynı olduğuna dikkat ediniz.

Cleaning tasarımında bulunan relation node’unun tanımı(**GW**:trashOccured); Environment ortamından miras aldığını belirtmek için özel bir syntax ile kullanılmıştır.

Aynı şekilde “cleaningProcessStart” planı içinde kullanılan relation node’u da bir üst model olan “Cleaning” yeteneğinden miras aldığını belirtir.(**Cleaning**:picked(grb))

Relation node tanımında noktalama işareti öncesi kullanılan tanım, kullanıcının miras aldığı ana modeli hatırlamasına yardımcı olmayı sağlarken, noktalama işareti sonrası bölüm kod dönüşümünde oluşturulacak olan sınıf ve method isimlerinde rol oynamaktadır.



 CleaningProcessStart planına ait bir görünüm.

**Kod ve model Dönüşümleri**

Verilen case ile ilgili modellemeler tamamlandıktan sonra uygulamanın sağ bölmesinde “Designer ToolBox” içerisinde bulunan açılır text(dropdown) bileşeninde, kullanıcının bu zamana kadar oluşturduğu proje isimleri görüntülenir. Model ve Kod çıktısı alınmak istenen proje adı seçildikten sonra “Export Code” ve “Export Modal” buttonları yardımıyla, proje ismi ile oluşturulacak bir zip dosyası içerisinde model ve kod dönüşümleri izlenebilir.



 Proje isimlerinin dropdown üzerinde seçiminden bir görünüm.

**Bağımsız bir uygulama oluşturmak için adımlar**;

1.Projects dizini içerisinde oluşturulacak projenin ismi ile yeni bir proje alt dizini oluşturulmalı.

2.Oluşturulan dizinin altına ilgili tasarım bileşenlerini oluşturacak dosyalar uzantıları ile oluşturulmalı (Add new file -> plan.**pln** gibi)

3.Oluşturulan tasarım dosyaları içerisinde “nodes” ve “edges bileşenlerine göre tasarımlar yapılıp, her aksiyondan sonra “save” işlemi yapılmalı.

4.Oluşturulan proje ile ilgili tasarım tamamlandıktan sonra kod ve model çıktısı almak için “Designer ToolBox” içerisinde “Projects” alanından dosya adı seçilerek butonlara tıklanmalıdır