ROS NEDİR?

ROS'un açınımı olan "Robot Operating System" robot işletim sistemi anlamına gelmektedir ancak ROS aslında bir işletim sistemi olmayıp birlikte bünyesinde robotik uygulamarda kullanılan yazılım kütüphanelerini barındırmaktadır. ROS'un kullanımının yaygınlaşmasıyla birlikte robotik uygulamalarda çalışmalar kolaylaşmış olup ilerleme kaydedilmiştir.

ROS aslında hali hazırda var olan işletim sisteminde bir ara yazılım görevi görmektedir. ROS1'in desteklendiği işletim sistemi sadece Linux iken ROS2 buna ek olarak Windows'ta da desteklenmektedir.

ÖNEMLİ ROS TERİMLERİ

ROS node: Amacı eş zamanlı olarak sadece bir görevi gerçekleştirmek olan en basit program. Başka bir deyişle aslında node en küçük işlemci birimidir. Bu noktada kendisine kısaca programcık diyebiliriz:)

ROS package: Paket aslında ROS'un en temel birimidir. Paket dediğimiz bu birimler bünyesinde node'ları ve diğer gerekli dosyaları barındırırlar. Bu dosyalar mesaj dosyaları, servis dosyaları, konfigürasyon dosyaları gibi birçok farklı dosya tipinde olabilir. ROS uygulamaları bu paketler temelinde ilerler.

ROS stack: Stack benzer amaca hizmet eden farklı paketleri bünyesinde bulunduran daha büyük bir birim olarak değerlendirebilir.

ROS workspace: ROS üzerinde yapılan tüm işlemler belli bir workspace üzerinde yapılmalı bu sayede ROS kullanıcının yazdığı dosyalar ve onların kaynakları için nereye bakması gerektiğini anlayabilsin.

ROS master: ROS node'larına erişebilmek ve aradaki gerekli iletişimi sağlayan bir server görevi görür. ROS master olmadan node'lar arasında iletişim sağlanamaz.

ROS topic: Topics node'lardan iletilerin geçmesini sağlayan istasyon görevi görür denebilir. Her topic'in kendisine özel bir adı vardır. Bu iletimde görev alan publisher(yayınlayıcı) ve subscriber(dinleyici) olmak üzere iki ana iletişim aracısı vardır. Bu aracılar topic isimleri uyuştuğu ve program izin verdiği sürece iletimi gerçekleştirirler. İsimlerinden de anlaşıldığı üzere publisher yayınlama yaparken subscriber ise onu dinleyen ve mesajı alan tarafta yer alır. Bu iletim publisher'dan subscriber'a doğru tek yönlü ve asenkron olarak gerçekleşir.

Publish: Data yayınlamak.

Subscribe: Yayınlanan datayı almak.

ROS publisher: Bir ROS topic'inde veriyi yayınlayan taraf.

ROS subscriber: Bir ROS topic'inde veriyi alan taraf.

Publishing and receiving süreci: Publisher node'u topic'ini ROS master'ına kaydettikten sonra o topic üzerinden mesajını yayınlamaya başlar. Aynı topic ile ilgili olan subscriber node'ları ise bu mesajı alır. Publisher ve subscriber node üzerinde tanımlanır ve bir node içinde birden çok kez atıfta bulunulabilir.

ROS service: Service senkron olan ve çift yönlü olarak iletimi sağlayabilen bir iletişim kanalı olarak tanımlanabilir. Bu süreçte yer alanlar ise service server ve service client'tır.

Service server: Service server "request" and "response" olmak üzere iki ana kısımdan oluşur. Request kısmı server'a girdi olarak verilir ve bunun sonucunda çıktı olarak bir response alınır. Request ve response'un her ikisi de mesaj türündedir. Service server'ı node içinde tanımlanır.

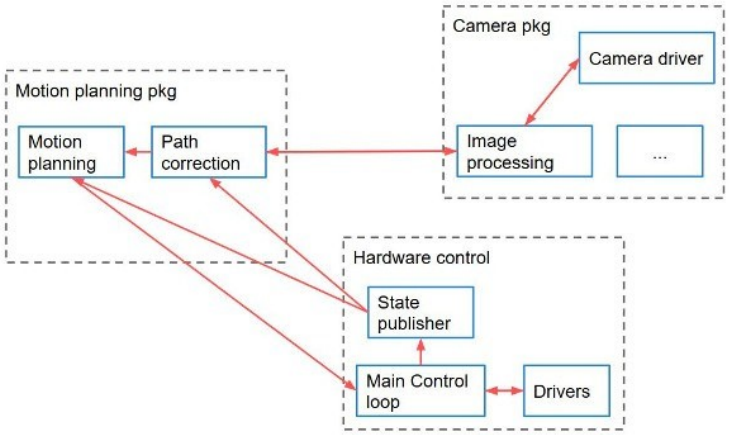
Service client: Service client, service server'ına request'te bulunan ve sonucunda response'u alan taraf olarak tanımlanabilir. Başka bir deyişle service server'ı bağlayıcı taraf, service client ise o server'ın müşterisi olan taraf olarak değerlendirilebilir.

ROS parameter: ROS parametreleri node içinde kullanılan ve o node'un konfigürasyonunu belirleyen parametrelerdir. Bu parametreler ilk başta varsayılan halleriyle gelir ve gerekirse üzerinde değiştirmeler yapılabilir. Bu parametreler üzerinde değişiklik yapabilmek özellikle gerçek hayat uygulamarında oldukça önemlidir.

Tüm bu terimlere ve tanımlarına bakıldığında aslında süreç hiyerarşik olarak şu şekilde ilerlemektedir:

- Üzerinde çalışılan ve bünyesinde gerekli dosyaları bulundurması istenen workspace
- Bu workspace altında içinde node'ları ve diğer konfigürasyon dosyalarını bulunduran package
- Bu package içinde node'lar
- Farklı node'lar arasındaki iletişim de topic ve service'lar aracılığıyla sağlanır.

Örnek uygulama şeması:



Burada bir robotik uygulamanın farklı parçaları görülmektedir.

Burada her paket altında farklı sensörleri ve birimleri barındırmaktadır.

Bunlar arasındaki iletişim ise iletişim kanalları aracılığıyla sağlanmaktadır.