XXII. Fiatal Műszakiak Tudományos Ülésszaka, 2017. Kolozsvár, 371–374. DOI: 10.33895/mtk-2017.07.85
http://hdl.handle.net/10598/29820

ROBOT OPERÁCIÓS RENDSZER A „ROS 2.0” LEHETŐSÉGEINEK TÜKRÉBEN

ROBOT OPERATING SYSTEM IN „ROS 2.0”

Szögi Gábor

Óbudai Egyetem Bejczy Antal iRobottechnikai Központ, Cím: 1034, Magyarország, Budapest, Bécsi út 96/B; Telefon: +36-1-6665700,, gabor.szogi@irob.uni-obuda.hu

Abstract

My research aims to provide insight into the robot operating system. It is presented, how it evolved to ROS and what has evolved over the years. ROS 2.0 in development will be presented during the system components, as well as the DDS system and its potential.

Keywords: *ROS, robotics surveillance, industry 4.0*

Összefoglalás

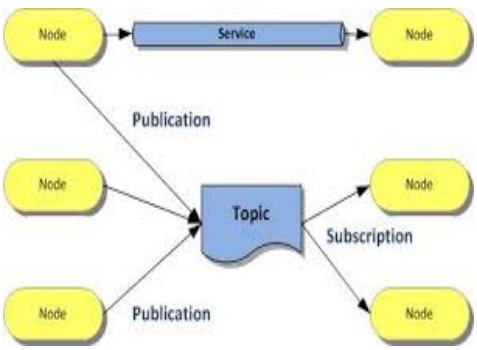
A kutatásom célja, hogy betekintést nyújtsak a robot operációs rendszerbe. Bemutatásra kerül, hogyan alakult ki a ROS, és milyen fejlődésen ment keresztül az évek során. A fejlesztés alatt lévő ROS 2.0 esetében bemutatásra kerülnek a rendszer alkotóelemei, valamint a DDS rendszer és a benne rejlő lehetőségek.

Kulcsszavak: *ROS, robitka felügyelet, industry 4.0*

1. ROS története

A Robot Operating System (ROS) egy nyílt forráskódú keretrendszer, amely saját könyvtár és eszközkészletével segítséget nyújt robot alkalmazások fejlesztésében. A meghajtókon keresztül a legkorszerűbb algoritmusokon át egy erős fejlesztési eszköz, amely segítséget nyújthat robotikai projektek megvalósításában. A ROS egy flexibilis keretrendszer, amely robot szoftverek fejlesztéséhez került kialakításra. Az eszköztárak és könyvtárak gyűjteménye egyezményesen került kialakításra, amelynek célja, hogy egyszerűsítse a fejlesztést a különböző robot platformokon keresztül. Sokan érezték szükségét a nyílt végű együttműködési rendszernek a robotikai kutatók körében és a cél elérése érdekében több projektet is indítottak. A 2000-es évek közepén a Stanford Egyetemen többek között a mesterséges intelligencia bevonásával két projektet indítottak. A STanford AI Robot (STAIR) és a Personal Robot (PR) projekt egy házon belül készített prototípus, amely egy rugalmas és dinamikus szoftver rendszert foglalt magában robotikai használatra szánva. 2007-ben Willow Garage, amely egy robotikával foglalkozó inkubátorház, jelentős forrásokat biztosított a projektek számára, hogy biztosítsa a koncepció megvalósítását. Ezek az erőfeszítések felpezsdítettek számtalan kutatócsoportot, akik idejükkel és szakértelmükkel hozzájárultak az alapvető szoftver csomagok kifejlesztéséhez, a ROS magjának megvalósításához. Az egész a szoftvert a nyílt forráskódú Berkeley Software Distribution (BSD) licence használatával fejlesztették és fokozatosan

terjedt el a robotikai kutató közösség körében. A ROS-t a kezdetektől több intézmény közreműködésével különböző robotokhoz fejlesztették, többek között a (PR) fejlesztőit a Willow Garage-ből. A ROS ökoszisztéma egyik erőssége, hogy az évek során az az eljárás alakult ki, hogy a forráskódok mindenki számára elérhetőek. Bármely fejlesztő csoport létre tudja hozni és elérhetővé teheti saját ROS kód gyűjteményét a saját szerverén. Ha a fejlesztő csoport úgy dönt, hogy nyilvánosan elérhetőek a forráskódok, akkor nem kellenek engedélyek a használatához. Ebből eredően a ROS ökoszisztémában több tízezer felhasználó vesz részt a hobby projektektől a nagy ipari automatizálási rendszereken keresztül. [1,3]



1. ábra. *ROS felépítése*

2. ROS 2.0

A projekt elindulásakor a fejlesztő közösség nem fektetett nagy hangsúlyt az alábbi felhasználói estekre, amelyek szükségessé tették a további fejlesztéseket:

- Több robot közös csoportokban: Bár a ROS lehetőségeset ad a multi robot rendszerek kialakítására, de nem alakult ki rá elfogadott szabvány, ezért könnyen hozzáférhető és feltörhető a ROS single-master struktúrája.
- Kisméretű beágyazott rendszerek: Kisméretű számítógépek alkalmazása, beleértve a mikro kontrollereket, hogy első osztályú része lehessen a ROS környezetnek.
- Valós idejű rendszerek: Támogathatóvá válik a valós idejű irányítás közvetlenül a ROS-ban, beleértve az „inter-process” és „inter-machine” kommunikációt, amelyhez elengedhetetlen a megfelelő operációs rendszer és a megfelelő hardver támogatás.
- Nem ideális hálózatok: Fontos cél, hogy a ROS alkalmassá váljon, amennyire lehetséges a hálózati kapcsolat elvesztése nélkül a megfelelő kommunikációs kapcsolat fenntartásához.
- Termelési környezet: Létfontosságú, hogy a ROS egy választási lehetőség maradjon a kutatási területen, de biztosítani kell a ROS alapú laborotóriumi prototípusok fejlesztését, hogy a való életben alkalmasak legyenek további felhasználásra.
- Előírt minták rendszerek építéséhez strukturáláshoz: A ROS-t a rugalmasság fémjelzi, miközben továbbra is fenntartja, hogy könnyen átlátható mintákat biztosít a fejlesztők számára.

A ROS magja egy anonim publish-subscribe middleware rendszerre épül szinte teljesen a nulláról építkezve. A fejlesztés időszakában amely 2007 óta tart, számos új technológia jelent meg, amelyek relevanciával bírnak a ROS 2.0 tekintetében az alábbi területeken:

- Zeroconf;
- Protocol Buffers;
- ZeroMQ
- Redis;
- WebSockets;
- DDS (Data Distribution Service).

Ma már lehetősége van ROS-szerű middleware rendszer építésére „off-the-shelf” nyílt forráskódú könyvtárak segítségével. Rendkívül hasznosak lehetnek ebből a megközelítésből többek között az alábbiak:

– Kevesebb kód kezelése, különösen a nem robotikai specifikus kódok tekintetében.
– Kihasználható funkciók azokban a könyvtárakban, amelyek kívül esnek azon, amit mi magunk készítettünk.
– Hasznos a folyamatos tökéletesítés terén, a mások által készített könyvtárak áttekintése és használata.
– Rámutathatunk a már meglévő termelő rendszerekre, amelyek a már meglévő könyvtárakra támaszkodhatnak. [2,3]

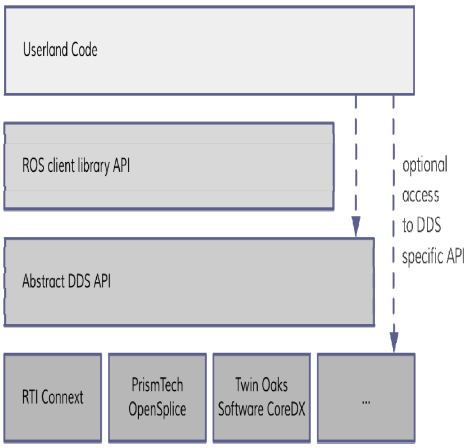
3. ROS 2.0 a DDS-ben

A cél, hogy elkészüljön a DDS a ROS 2.0 implementációs részleteivel. Ez azt jelenti, hogy az összes DDS specifikus API-t és üzenet definíciót el kellene rejteni. A DDS biztosítja a felfedezést, üzenet definíciót, üzenet sorba rendezést és a publis-subscribe szállítását.

3.1. Mi a DDS?

A DDS (Data Distribution Service) biztosítja a publish-subscribe szállítást, amely nagy mértékben hasonlít a ROS publish-subscribe szállításához. A DDS az IDL (Interface Description Language) által meghatározott nyelvet használja, amely az OMG (Object Management Group) által lett kialakítva az üzenetek definiálására. és sorba rendezésére. A DDS-nek a „kérés-válasz” stílusú szállítása olyan lenne, mint a ROS szolgáltató rendszere, amelyet DDS-RPC-nek neveztek el.

Az alapértelmezett felfedező rendszer a DDS által biztosított, amely szükséges a DDS purblish-subscribe rendszer szállításához, amely egy elosztott felfedező rendszer. Ez lehetővé teszi bármely két DDS program közötti kommunikációt anélkül, hogy ROS master-re lenne szükség. Ez teszi a rendszert még hibatűrőbbé és rugalmassá. Nem szükséges a dinamikus felfedező mechanizmusok használata, azonban a többszörös DDS gyártók lehetőséget biztosítanak a statikus felfedezésre. [4]



2. ábra. *DDS és ROS API layout*

4. ROS 2.0 alkotó elemei

Bár nem lehet biztosítani egy átfogó listát, hogy mit tartalmaz a ROS ökoszisztéma, ettől függetlenül azonosítani lehet az alapvető részeit, amelyek a funkcionalitás, műszaki előírások és a minsőség érdekében hozzájárulhatnak a ROS projekthez. [1]

4.1. Kommunikációs infrastruktúra

A legalsó szinten a ROS egy üzenet passing interfészt kínál, amely a folyamatok közötti kommunikációt biztosítja, amit gyakran middleware-nek neveznek.

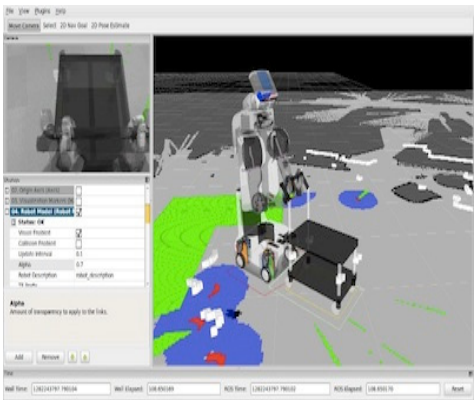
4.2. Robotspecifikus jellemzők

A fő middleware elemeken felül a ROS közös robot-specifikus könyvtárakat és eszközöket biztosít, amellyel könnyen és gyorsan tudunk robotot összeállítani és beállítani. Néhány robot specifikus képesség, amelyet a ROS biztosít:

Robot geometria könyvtár, robot leíró nyelv, diagnosztika, lokalizáció, térképezés, navigáció

4.3. Eszközök

Az egyik legerősebb jellemzője a ROS-nak az erős fejlesztési eszközkészlet. Ezek az eszközök támogatják az önelemzést, a hibakeresést, a plottingot és a vizualizációt. A rendszer alapjául szolgáló publis-subscribe mechanizmus lehetővé teszi, hogy spontán módon elemezhetővé válnak a rendszerbe áramló adatok, így könnyebbé válik a hibaellenőrzés. A ROS eszközök kihasználják az önvizsgálati képességeket, a kiterjedt grafikus könyvtárakat és a parancssoros segédprogramokat, így leegyszerűsödhet a fejlesztés és hiba-keresés.



3. ábra. *rviz layout*

5. Következtetések

A kutatás eredményeként megállapít-ható, hogy a ROS egy hatalmas projekt, rengeteg közreműködő fejlesztő együttműködésével.

Sokan érezték szükségét a nyílt végű együttműködési rendszernek a robotikai kutatók körében és a cél elérése érdekében több projektet is indítottak. A ROS-t használva könnyű újra felhasználható kódot készíteni. Az ROS esetében több kisebb alkalmazás fut egyidejűleg – ezeket node-oknak nevezzük, amelyek közül a legfontosabb node a 'roscore'. Ez a node biztosítja az alapvető funckiókat, és a node-ok kommunikációját. A különböző node-ok a 'roscore' node-t keresik fel, azáltal ismerik fel környezetüket Még mindig rengeteg kérdés merül fel a DDS hasznosítása esetében.

Köszönetnyilvánítás

Ezúton szeretnék köszönetet mondani az Óbudai Egyetem Robottechnikai Szakkollégium részére, amelynek tagjaként számos szakmai és anyagi támogatást kaptam jelen szakmai tudományos publikációm elvégzéséhez. A szakkollégium kiemelt segítséget nyújt az Óbudai Egyetem tehetséges hallgatóinak és doktoranduszainak.

Szakirodalmi hivatkozások

[1] http://www.ros.org/
[2] http://design.ros2.org/
[3] A. Koubaa, *Robot Operating System* (ROS): The Complete Reference. Springer, 2016.
[4] http://design.ros2.org/articles/ros_on_dds.html