Definitie: <<poza ros>>

ROS este un meta-sistem de operare open-source pentru roboți.Acesta oferă pe lângă serviciile standard oferite de un sistem de operare ( de exemplu abstractizare hardware,management al pachetelor și al proceselor), și funcționalități de nivel înalt (date de baze centralizate, comunicări sincrone și asincrone, un sistem de configurare pentru roboți, instrumente și librării pentru obținerea, crearea scrierea și rularea de cod specific).

Motivul alegerii:

Înaintea apariției acestui sistem orice proiectant de roboți sau cercetător din domeniu era nevoit să isi aloce o cantitate mare de timp pentru a putea realiza părțile componente necesare funcționării robotului, de la sistemul embedded al robotului până la fiecare componentă hardware. Deși existau destule materiale pentru realizarea teoretică a suportului de bază al proiectului, într-un cuvânt prototipul, reunirea acestora sub un tot unitar de hardware și software necesită de asemenea cunoștințe avansate în ingineria mecanică, electronică, precum și în programarea embedded. Așadar, robotica necesită dobândirea și aplicarea de cunoștințe din diverse domenii, fapt ce este greu de realizat de către un singur individ.

Ideea principala pe care se bazează un Sistem de Operare pentru roboti este aceea de a evita „reinventarea rotii”, și aducererea unei abstractizări hardware pe care să se folosească funcționalități standardizate, cum se întâmplă cu sisteme de operare convenționale pentru PC-uri, de aici trăgându-se analogia de nume.

Un alt beneficiu adus unui sistem de operare pentru roboți precum ROS, este combinarea expertizei provenite din diferite discipline. Un sistem de Operare propriu-zis înseamnă:

- Gestionarea hardware-ului prin scrierea de driver-e.

- Gestionarea memoriei și al proceselor.

-Gestionarea situațiilor de concurență, paralelism și fuziune a datelor;

- Asigurarea abstractizarii algoritmilor relaționari, deschizând astfel oportunități pentru folosirea inteligenței artificiale.

Sistemul de Operare

Variantele stabile de ROS rulează momentan pe platforme bazate pe Unix. Software-ul pentru ROS este testat în principal pe sistemele Ubuntu și MAC OS X, iar variantele pentru alte sisteme, ca de exemplu Fedora, Gento, Arch Linux, chiar și pentru Microsoft Windows, sunt oferite de către comunitatea ROS, aceste versiuni fiind încă în stadiul de continuă dezvoltare.

Programarea în ROS:

ROS este independent de limbajul de programare utilizat. În momentul scrierii acestei lucrări sunt existente trei librării principale pentru ROS, ce fac posibilă programarea ROS în Python, Lisp sau C++. Pe lângă acestea mai există alte 2 librării ce fac posibila programarea în Java sau Lua, acestea fiind însă în stadiul experimental.

Conceptele esentiale:

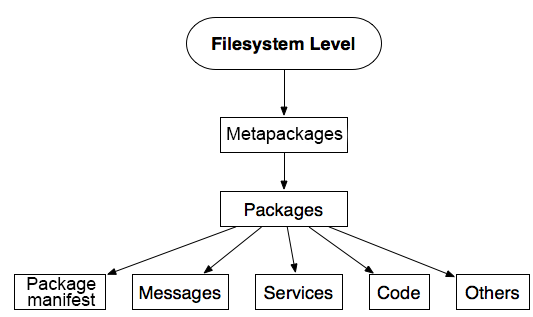
Conform dezvoltatorilor ce au dus la apariția ROS, acesta are trei nivele de concepte:

- Nivelul Sistemului de Fișiere;

- Nivelul Graficului de Calcul;

- Nivelul Comunitătii;

Nivelul Sistemului de Fișiere( din eng. : „filesystem”) este cel mai de bază nivel pentru crearea blocurilor necesare operării funcționării elementelor în ROS.(fig)



<https://hub.packtpub.com/ros-filesystem-levels/> <<stolen>>

Acesta acoperă conceptele legate de resursele ROS aflate pe disc, precum:

-pachetul: acesta fiind principala unitate de organizare a software-ului în ROS. Un pachet conține executabile, librării externe, seturi de date, fișiere de configurare și de creare a acestor pachete.În general pachetele ROS urmăresc principiul ”Goldilocks”: destulă funcționalitate pentru a fi folositoare, dar nu prea multă astfel încat acestea să nu fie prea încărcate și greu de utilizat de către alte software-uri.Pachetele sunt ușor de creat de mână, sau folosind un instrument precum „catkin\_create\_pkg”. În esență un pachet este pur și simplu un director descendent din varibila de mediu „ROS\_PACKAGE\_PATH” ce conține un fișier „package.xml”. Pachetele sunt unitatea cea mai de bază de creat și distribuit.

- metapachetul: metapachetele sunt pachete specializate care au ca singur scop reprezentarea unui grup de pachete ce sunt asociate între ele.

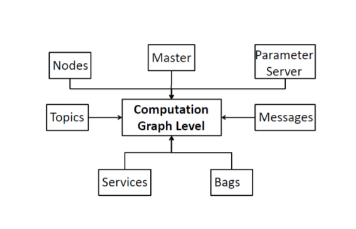
- manifestul pachetului: Manifestele sub forma de fișiere de tip XML denumite „package.xml” ce trebuie inclus în directorul oricărui pachet ce se conformează cu standardul „catkin”. Fișierul definește proprietătile pachetului, ca de exemplu numele acestuia, o descriere, informația despre licența folosită, dependințele necesare și alte meta-informații, precum pachetele exportate.

- depozitul (din eng. „repository”): O colecție de pachete ce împărtășesc același VCS. Pachetele ce împărtășesc un VCS sunt eliberate sub aceeași versiune.

- tipurile de mesaje (msg) : ROS folosește un limbaj simplificat de descriere al mesajelor ce descriu valorile datelor transmise. Acestă descriere ușurează generarea automată de cod sursă pentru tipul de mesaj pentru limbaje de programare specifice. Descrierea mesajelor sunt stocate în fișiere de tipul „mgs” în subdirectorului „msg/” al unui pachet ROS. Tipurile de mesaj sunt apelate folosind „package\_reosurce\_names”. Spre exemplu, fișierul „geometry\_msgs/msg/Twist.msg” este apelat sub numele de „gemetry\_msgs/Twist”.

- tipurile de servicii(srv) : Pentru descrierea tipurilor de servicii, ROS se folosește de un limbaj simplificat de descriere al serviciilor(„srv”) ce se bazează direct pe formatul ROS msg pentru a permite comunicarea be bază de „cerere/răspuns”. Descrierea serviciilor sunt stocate în fișiere de tipul „.srv” în subdirectorului „srv/” al unui pachet ROS. Tipurile de servicii sunt apelate folosind la fel ca mesajele, „package\_reosurce\_names”. Spre exemplu, fișierul „robot\_srvs/srv/SetJointCmd.srv” este apelat sub numele de „robot\_srvs/SetJointCmd”.

Al doilea nivel este cel al Graficului de Calcul, ce reprezintă o rețea de tipul peer-to-peer al proceselor ROS ce prelucrează împreună datele. Conceptele de bază al acestui nivel sunt „nodurile”, „master-ul”, „serverul de parametrii”, „mesajele”, „serviciile”, „topicurile” și „bag-urile”, acestea împreună asigurând fluxul de date către grafic sub diferite forme.



<https://www.theseus.fi/bitstream/handle/10024/106340/Jusuf_Fiki.pdf?sequence=1> <<taken from here>>

Toate aceste concepte sunt implementate în depozitul „ros\_comm”.

- Nodurile sunt procese ce realizează calcule. ROS este creat pe baza principului modularității până la cel mai de jos nivel, astfel ca orice sistem de control al roboților are în alcătuirea sa mai multe noduri. Un nod poate controla un telemetru lase, altul realizează localizarea, un altul poate controla motoarele roților, altul poate furniza o vedere grafică asupra sistemului, e.t.c. Orice nod poate fi scris folosind o librărie client specifica ros, precum „roscpp” sau „rospy”.

- Masterul ROS asigură numirea și serviciile de înregistrare a nodurilor în sistemul ROS. Acesta este implementat folosind protocolul XMLRPC pentru realizarea comunicării dintre noduri prin transmiterea de informații și gestionarea de servicii.

- Serverul de Parametri permite stocarea datelor cu o anumită cheie de referință într-o locație centrală. Acesta este inclus în Master sub forma unui protocol XMLRPC. Datele stocate sunt de obicei parametrii de configurare și alte date statice ce sunt vizibile pentru toate nodurile.

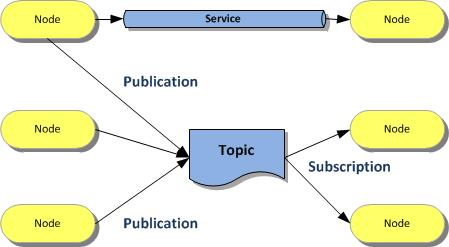
- Mesajele sunt structuri compuse din tipuri de date. Aceste tipuri pot fi primitive, precum întregi, cu virgulă mobilă, valori logice, etc, dar și compuse din alte mesaje, această formă se aseamăna cu vectorii de tipuri primitive recent menționate. Definiția mesajelor a fost amintită în subcapitolul anterior.

- Topicele ( din eng. „topics”) reprezintă sistemul de transport asincron pentru mesaje folosind un sistem de „publicare/abonare”. Un nod poate transmite un mesaj publicându-l pe un topic. Topicul este un nume ce este folosit pentru a identifica conținutul mesajului. Un nod ce este interesat de un tip specific de informații se va abona la topicul corespunzător. Numărul de noduri ce publică sau se abonează la un anumit topic poate fi mai mare de unul, și de asemenea, un nod poate publica și se abona la mai multe topice. În general nodurile ce interacționeaza cu un topic nu sunt conștiente de prezența fiecăruia, mergându-se pe ideea că producția de informații nu trebuie să fie strâns legată de consumarea acesteia. Topicele se pot asocia ca idee cu fluxurile RRS de date.

Modelul comunicare de tipul de publicare/abonare asincron de tipul many-to-many este unul foarte flexibil, însă nu este adecvat pentru comunicarea sincronă dintre două noduri.

-Serviciile sunt definite ca fiind o structura de perechi de mesaje, unul fiind pentru a trimite o cerere și altul pentru a primi răspunsul. Acestea funcționează în modul următor: un nod furnizor oferă un serviciu sub un nume și un client folosește serviciul trimițând o cerere sub forma unui mesaj și asteaptă un răspuns. Librăriile client ROS prezintă în general această interacțiune programatorului ca și cum ar fi fost un aper de procedură solitar.

Diferența dintre cele 2 concepte se poate observa în figura de mai jos:



- Bag-urile reprezintă un format pentru salvarea și redarea datelor din mesajele ROS. Acestea sunt un mecanism important de stocare a diferitelor date, ca de exemplu cele de la un senzor, ce sunt în general greu de colectat, dar sunt necesare pentru dezvoltarea și testarea de algoritmi.

Al treilea nivel ce este la fel de important ca primele două este acela de Comunitate. Conceptele create de acest nivel reprezintă resursele ROS ce permit comunităților separare să facă schimb de informații și software.

- Distribuțiile ROS sunt colecții de depozite versionate ce pot fi instalate de oricine. Acestea joacă un rol similar cu cel al distribuțiilor Linux: ele ușurează procedeul de instalare al unei colecții de software și țin cont de versionarea corectă a setului de software. Odată ce o distribuție este lansată se încearcă limitarea apariției de patch-uri pentru rezolvarea erorilor în profida dezvoltatorilor ce vor să creeze cod bazat pe distribuția folosită.

Există multe distribuții apărute de-a lungul anilor, denumirile acesora fiind în ordine alfabetică. Ultimele distributii disponibile până la momentul publicării lucrării de licență de față, sunt:

-ROS Box Turtle, publicată pe 02 Martie 2010

-ROS C Turtle, publicată pe 02 August 2010

-ROS Diamondback, publicată pe 2 Martie 2011

-ROS Electric Emys, publicată pe 30 August 2011

-ROS Fuerte Turtle, publicată pe 23 Aprilie 2012

-ROS Groovy Galapagos, publicată pe 31 Decembrie 2012

-ROS Hydro Medusa Turtle, publicată pe 04 Septembrie 2013

-ROS Indigo Igloo, publicată pe 22 Iulie 2014

-ROS Jade Turtle, publicată pe 23 Mai 2015

- ROS Kinetic Kame, publicată pe 23 Mai 2016

-ROS Lunar Loggerhead, publicată pe 23 Mai 2017

-ROS Melodic Moreina, publicată pe 23 Mai 2018;

<<take this from here>>Deoarece distribuția ROS Kinetic Kame vine cu un suport extins la 5 ani și are pachetele necesare stabile la momentul începerii acestei lucrări, am ales-o pe aceasta ca fiind cea pe care se bazează lucrarea de față.(Cu posibilitatea actualizării lucrării la distribuția ROS Melodic Moreina).<</take this from here>>

- Depozitele, menționate și la nivelul Sistemului de fișiere, sunt modul în care dezvoltatorii( individuali sau instituții ) pot creea și distribui propriile lor componente software pentru roboți.

- Comunitatea Wiki ROS este principalul forum pentru documentarea informațiilor despre ROS. Aceasta urmează modelul site-ului wikipedia.org ce permite înscrierea utilizatorilor și contribuirea cu propria documentație, corectarea sau actualizarea, scrierea de tutoriale, și altele.

- Ca un orice sistem open-source și ROS este vulnerabil la erori, iar acestea pot fi raportate de către utilizatori prin Sistemul de Tichetare al Bug-urilor.

- Mailing List, ROS Answers și Blog sunt alte trei resurse prin care comunitatea ROS poate face schimb de informații relative la software-ul ROS.

Pe lângă aveste trei nivele principale de concepte ROS definește alte două tipuri de nume: Package Resource Names și Graph Resources Names.

- Primul amintit oferă o strucrură ierarhică de denumire ce este folosită pentru toate resursele dintr-un Grafic de Calcul ROS, printre care Nodurile, Parametrii, Topicele și Serviciile. Aceste nume sunt foarte importante în ROS pe măsură ce gradul de complicare al sistemelor din ROS se poate mări, astfel că este necesară înțelegerea folosirii denumirilor.

Numele Resurselor din Grafic sunt un important mecanism din ROS pentru oferirea proprietății de încapsulare.Fiecare resursă este definită sub un „namespace”, ce poate fi comun cu alte resurse. În general resursele pot crea alte resurse în interiorul propriului „namespace” și pot accesa aceste resurse create, din interiorul sau din afara acestuia. Pot fi făcute conexiuni între resurse aflate în diferite „namespace”-uri, dar asta se poate realiza prin integrarea codului în afara „namespace”-ului. Această încapsulare izolează diferite porțiuni din sistem din a lua resurse numite greșit sau a prelua nume făra acord.

Se poate de asemenea realiza remaparea oricărui nume dintr-un nod ROS atunci când acesta este lansat folosind linia de comandă.

- Numele Resuselor din Pachete sunt folosite în conceptele ROS de la nivelul Fișierelor de Sistem pentru a simplifica procesul de referire la fișiere și tipuri de date de pe disc. Acestea sunt foarte simple, folosind numele pachetului în care se află resursa plus numele propriu-zis al acesteia. Spre exemplu, numele „std\_msgs/String” se referă la tipul de mesaj „String” din pachetul „std\_msgs”. Astfel majoritatea fișierelor folosite în ROS ce sunt apelate folosind Numele Resuselor din Pachete includ tipuri de mesaje, servici și de topice.

Numele Resuselor din Pachete se aseamănă foarte mult cu căile de fișiere, cu deosebirea că sunt cu mult mai scurte., fapt datorat abilității ROS-ului de a putea localiza pachete de pe disc și de a face presupuneri asupra conținutului acestora. Spre exemplu, descrierile de mesaje sunt salvate mereu în subdirectorul „msg” și au extensia „.msg”, asftel că „std\_msgs/String” este prescurtarea de la „<calea\_către\_pachetul\_std\_msgs>/msg/String.msg”. În mod similar, nodul de tipul „foo/bar” este echivalentul căutării unui fișier denumit „bar” în pachetul „foo” ce are dreptul de a fi executat.