Robotic System Specification Methodology Based on Hierarchical Petri Nets

Maksym Figat

Warsaw University of Technology, Poland
Institute of Control and Computation Engineering

Office national d'études et de recherches aérospatiales (ONERA), France
02.12.2021

European Funds Knowledge Education Development
Republic of Poland
NAWA Polish National Agency for Academic Exchange
European Union European Social Fund

Plan of presentation

Motivation

- International Federation of Robotics (IFR) anticipates that number of service robots will **triple** in 2023 compared to 2019 [1],
- significant **increase** in the number of new robotic system implementations,
- development of software of a robotic system is a **challenge**,
- tools **facilitating the design** of such systems are in high demand,
- so far, neither a **universal method** for designing robotic systems nor a **universal architecture** for robotic systems has been developed [2].

Motivation II

Approaches to system development usually provide:

- **freedom of choice**,
- software **modules** supplemented with communication mechanisms [3],

However, they do not provide:

- guidelines,
- rules,

on **how to develop** a system that has to execute the required task, which is **evident** for Robot Operating System (ROS) or Open RObot COntrol Software (OROCOS).

The quality of the resulting systems depends primarily on:

- skills,
- experience of the designer.

Motivation III

European SPARC project [4] indicates that the **model-based approach** (i.e. Model Driven Engineering (MDE)) has the **potential to become popular and play an important role in the design of robotic systems**.

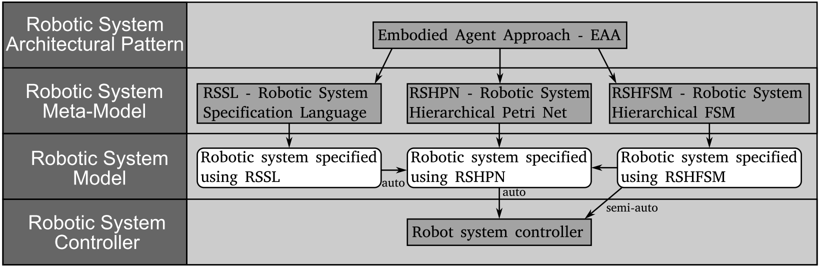
MDE:

- Introduces a **meta-model** and **design patterns**,
- Gives appropriate **balance** between guidance and flexibility ensuring appropriate system structure and operation [4].

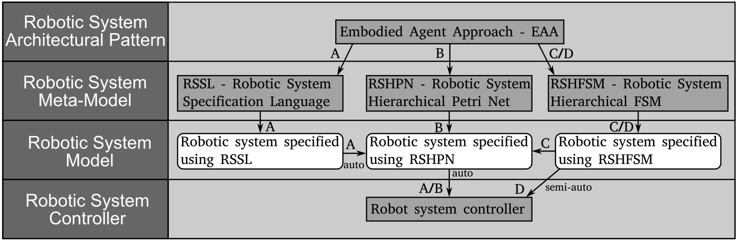
Main goal

Our objective is to develop a **methodology** for designing robotic systems based on **concepts from the field of robotics** and based on the **MDE approach**.

Robotic System Specification Methodology



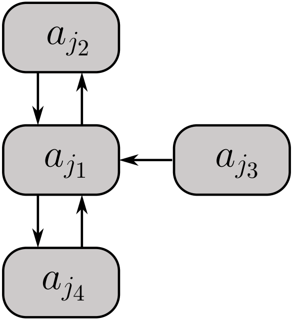
Robotic System Specification Methodology



- A - article in preparation,
- B - Access 2020 [5], ICRA 2019 [6],
- C - Automation 2019 [7],
- D - RoMoCo 2017 [8], JINT 2019 [9], Automation 2018 [10].

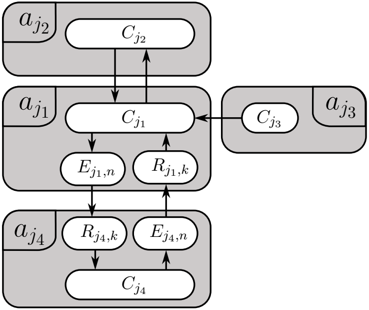
Structure - system decomposition into agents

A)



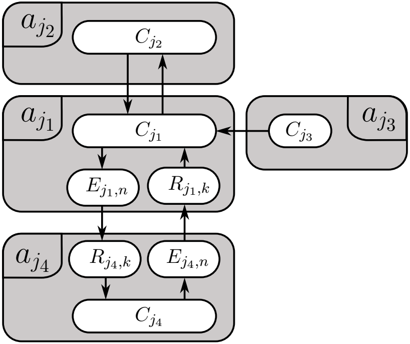
$\Rightarrow$

B)



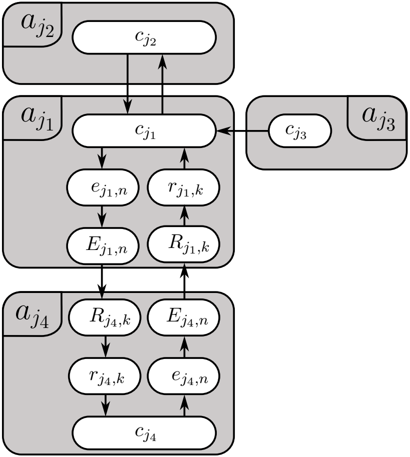
Structure - system decomposition into agents

B)

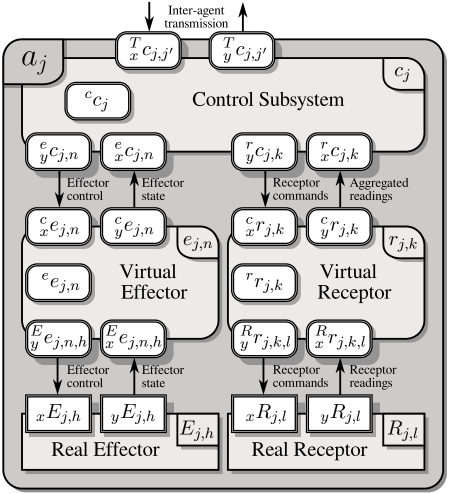


$\Rightarrow$

C)

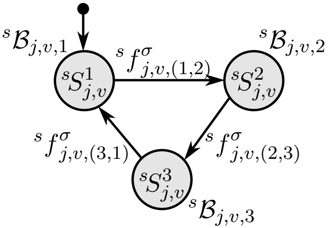


Structure - system decomposition into cooperating agents

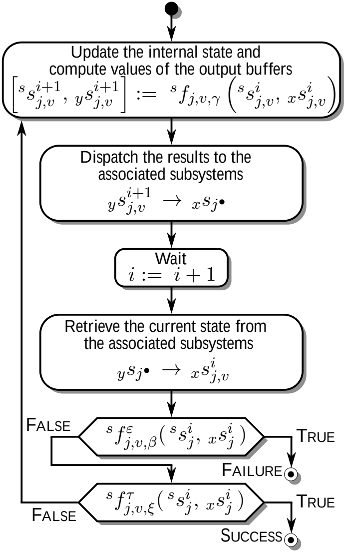


Activity of robotic system $\mathcal{RS}$

- $\mathcal{RS}$ performs a task $\mathcal{T}$,
- $\mathcal{T}$ is composed of tasks $\mathcal{T}_j$ performed by a_j,
- $\mathcal{T}_j$ is composed of tasks $\mathcal{T}_{j,v}$ performed by $s_{j,v}$,
- the execution of task $\mathcal{T}_{j,v}$ involves performing and switching between behaviours ${}^s\mathcal{B}_{j,v,\omega}$,
- subsystem $s_{j,v}$ **switches between behaviours** based on initial condition ${}^sf^{\sigma}_{j,v,\alpha}$.



Activity of behaviour ${}^{s}\mathcal{B}_{j,v,\omega}$



Transition function - decomposition

$$[{}^{s}s^{i+1}_{j,v},\ {}^{s'}_{y}s^{i+1}_{j,v,v'}] := {}^{s}f_{j,v,\gamma}\left({}^{s}s^{i}_{j,v},\ {}^{s'}_{x}s^{i}_{j,v,v'},\ {}^{s''}_{x}s^{i}_{j,v,v''}\right)$$

Canonical decomposition

Decomposition of ${}^{s}f_{j,v,\gamma}$ with respect to the **destination of the calculated data**: i.e. output buffer or internal memory buffer, a partial transition functions are created:

- ${}^{s'}_{y}s^{i+1}_{j,v,v'} := {}^{s,s'}f_{j,v,\gamma,\psi}\left({}^{s}s^{i}_{j,v},\ {}^{s'}_{x}s^{i}_{j,v,v'},\ {}^{s''}_{x}s^{i}_{j,v,v''}\right)$,
- ${}^{s}s^{i+1}_{j,v} := {}^{s,s}f_{j,v,\gamma,\psi'}\left({}^{s}s^{i}_{j,v},\ {}^{s'}_{x}s^{i}_{j,v,v'},\ {}^{s''}_{x}s^{i}_{j,v,v''}\right)$,

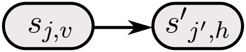
Decomposition by data access

Partial transition function ${}^{s,s'}f_{j,v,\gamma,\psi}$ is decomposed due to the **availability of input data**, an overloaded transition function is created accepting **various combinations** of input buffers.

Communication

- Inter-agent communication (using transmission buffer, and using environment),
- Intra-agent communication

Communication occurs always between a pair of subsystems, e.g. $s_{j,v}$ and $s'_{j',h}$:



Sender acting in mode:

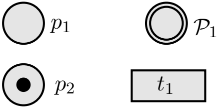
- blocking $\equiv$ B,
- blocking with timeout $\equiv$ BT,
- non-blocking $\equiv$ NB

Receiver acting in mode:

- blocking,
- blocking with timeout,
- non-blocking

For a pair of subsystems 9 different combinations of communication models are possible!

HPN



Transition associated with:

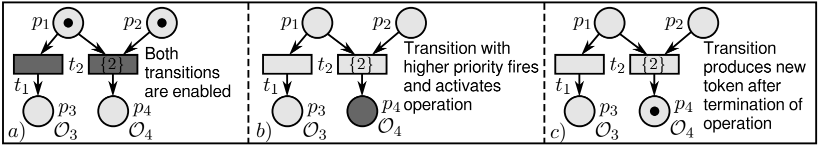
- condition – $\mathcal{C}$,
- priority – $\mathcal{P}r$,
- timeout

Place associated with:

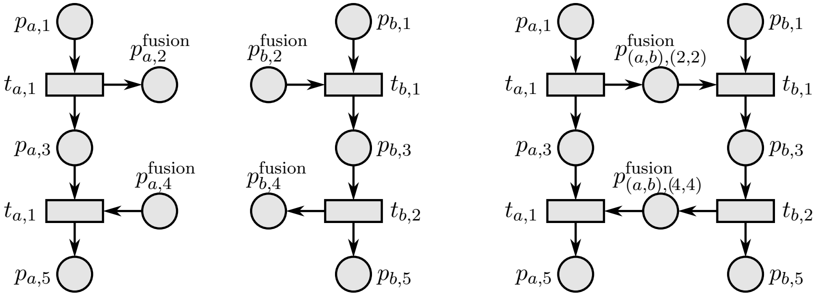
- operation – $\mathcal{O}$,
- number of tokens

Page associated with:

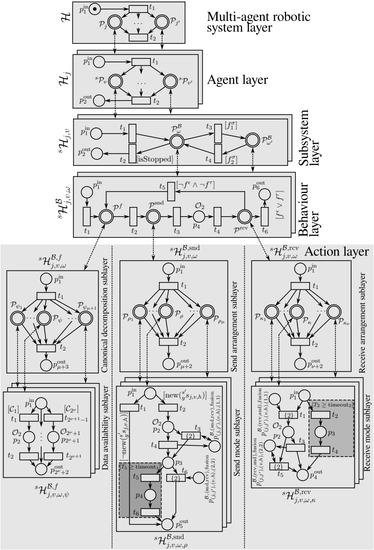
Petri net (HPN)



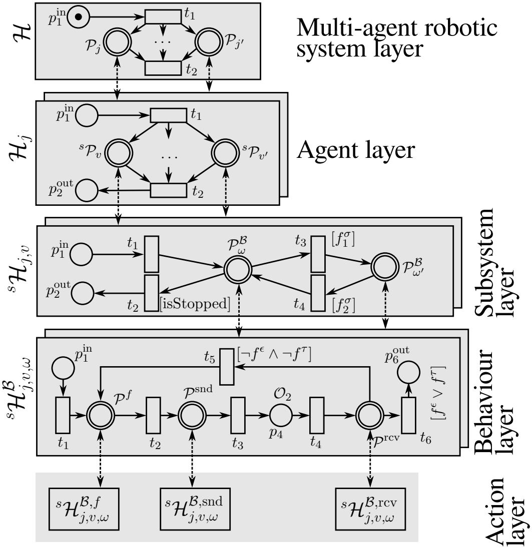
Place fusion



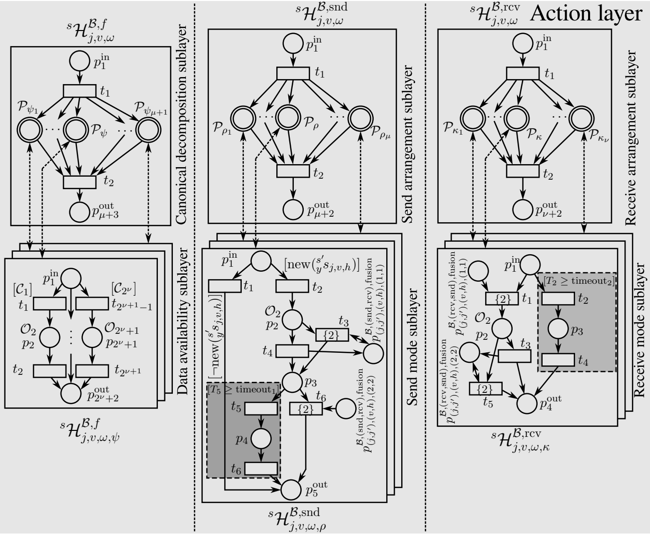
Robotic System HPN (RSHPN) meta-model



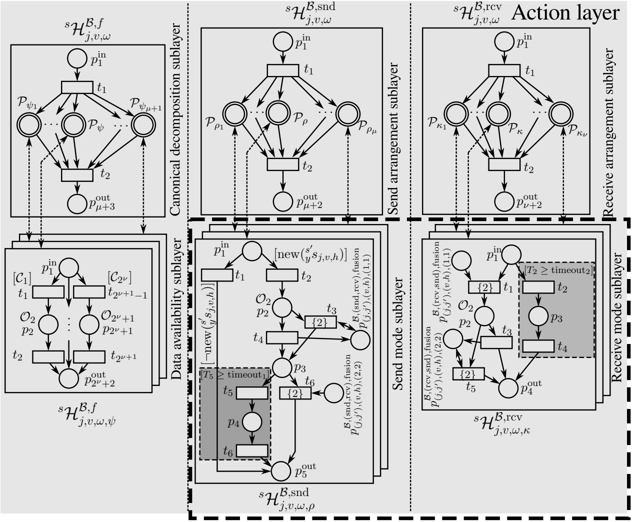
RSHPN modelling a robotic system



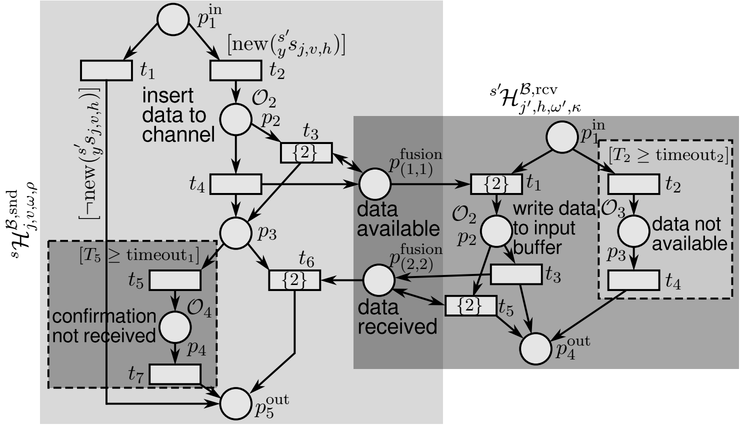
RSHPN - Action layer



Communication model



General communication model



Communication model depends on timeouts

Communication model		Receive: $\text{timeout}_2 = 0$	Receive: $\text{timeout}_2 = \infty$	Receive: $0 < \text{timeout}_2 < \infty$
Send	$\text{timeout}_1 = 0$	NB-NB	NB-B	NB-BT
Send	$\text{timeout}_1 = \infty$	B-NB	B-B	B-BT
Send	$0 < \text{timeout}_1 < \infty$	BT-NB	BT-B	BT-BT

where:

- NB – non-blocking mode,
- B – blocking mode,
- BT – blocking mode with timeout.

RSHPN analysis

RSHPN properties analysis

RSHPN complexity analysis

RSHPN properties analysis - Idea

- decomposition of RSHPN into layers, and panels,
- analysis of Petri nets from panels (reduction methods used),
- analysis of Petri nets associated with communication and nets composed of multiple layers,
- three types of PN: trivial, unchangeable, user defined,

Properties verification:

- 1-boundedness, safety,
- lack of deadlocks,

Analysis methods:

- graphical (for trivial nets),
- reachability graph,
- methods based on place and transition invariants

RSHPN COMPLEXITY ANALYSIS

Complexity depends on number of places/transitions/edges

The higher the complexity, the more difficult/longer it is to develop a Petri net. Also, the greater the error prone.

Number of places/pages in $\mathcal{H}$:

$$\text{places}(\mathcal{H}) = \sum_{a_j \in \hat{a}} \Big(\sum_{s_{j,v} \in \hat{s}_j} |{}^{s}\hat{\mathcal{B}}_{j,v}| \cdot \Big(2^{|{}_{x}\hat{s}_{j,v}|} \cdot \Big(|{}_{x}\hat{s}_{j,v}| + 1 \Big) + 8 \cdot |{}_{x}\hat{s}_{j,v}| + \\ + 10 \cdot |{}_{y}\hat{s}_{j,v}| + 16 \Big) + 3 \cdot |\hat{s}_j| \Big) + 3 \cdot |\hat{a}| + 1 \tag{1}$$

Number of transitions in $\mathcal{H}$:

$$\text{transitions}(\mathcal{H}) = \sum_{a_j \in \hat{a}} \Big(\sum_{s_{j,v} \in \hat{s}_j} \Big(|{}^{s}\hat{\mathcal{B}}_{j,v}| \cdot \Big(\Big(|{}_{y}\hat{s}_{j,v}| + 1 \Big) \Big(2^{|{}_{x}\hat{s}_{j,v}| + 1} \Big) + 4 \cdot |{}_{y}\hat{s}_{j,v}| + \\ + 3 \cdot |{}_{x}\hat{s}_{j,v}| + 10 \Big) + |{}^{s}\hat{f}^{\sigma}_{j,v}| \Big) + 2 \cdot |\hat{s}_j| \Big) + 2 \cdot |\hat{a}| + 2 \tag{2}$$

Number of edges in $\mathcal{H}$:

$$\text{edges}(\mathcal{H}) = \sum_{a_j \in \hat{a}} \Big(\sum_{s_{j,v} \in \hat{s}_j} \Big(|{}^{s}\hat{\mathcal{B}}_{j,v}| \cdot \Big(2^{|{}_{x}\hat{s}_{j,v}| + 2} \cdot \Big(|{}_{y}\hat{s}_{j,v}| + 1 \Big) + 21 \cdot |{}_{y}\hat{s}_{j,v}| + \\ + 16 \cdot |{}_{x}\hat{s}_{j,v}| + 20 \Big) + 2 \cdot |{}^{s}\hat{f}^{\sigma}_{j,v}| \Big) + 6 \cdot |\hat{s}_j| \Big) + 4 \cdot |\hat{a}| + 1 \tag{3}$$

RSHPN complexity analysis

Nr	$\|\hat{a}\|$	$\|\hat{s}_j\|$	$\|{}^s\hat{\mathcal{B}}_{j,v}\|$	$\|{}_x\hat{s}_{j,v}\|$	$\|{}_y\hat{s}_{j,v}\|$	places($\mathcal{H}$)	transitions($\mathcal{H}$)	edges($\mathcal{H}$)	Total
1	1	1	1	1	1	45	32	86	163
2	1	1	1	5	5	125	76	266	467
3	5	1	1	1	1	221	152	426	799
4	5	1	1	5	5	621	372	1326	2319
5	5	5	1	1	1	1041	712	2046	3799
6	5	5	5	1	1	4841	3312	9546	17699
7	5	5	5	5	5	14841	8812	32046	55699

Note:

For system no. 7, 55699 elements must be created!

Conclusion:

RSHPN $\mathcal{H}$ must be generated **automatically**! $\Rightarrow$ RSSL

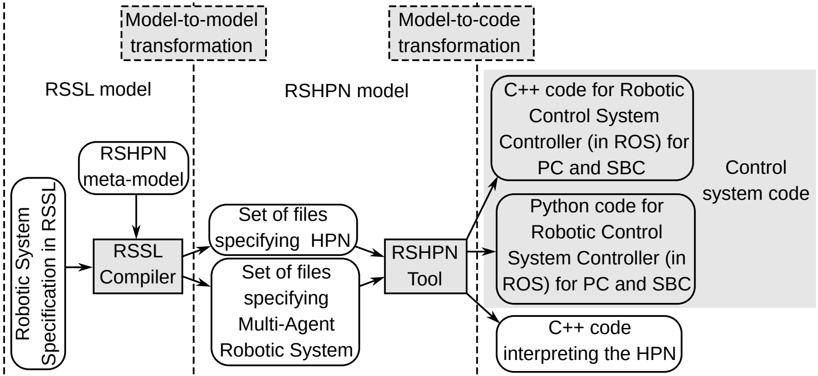
Robotic System Specification Language

- a domain language for specifying multi-agent robotic systems,
- based on concepts derived from robotics, in particular the embodied agent,
- RSSL specifies both the structure and the activity of a robotic system,
- RSSL is specified using a context-free grammar expressed in BNF form.

RSSL enables RSHPN meta-model parameterization

Layer/Sublayer	Parameters
Multi-agent robotic system	$\hat{a}$
Agent	for each $a_j \in \hat{a}$: $\hat{s}_j$
Subsystem	for each $s_{j,v} \in \hat{s}_j$: ${}^{s}\hat{\mathcal{B}}_{j,v}$, ${}^{s}\hat{f}^{\sigma}_{j,v}$, ${}_{y}\hat{s}_{j,v}$, ${}_{x}\hat{s}_{j,v}$ and ${}^{s}\mathcal{T}_{j,v}$
Behaviour	for each ${}^{s}\mathcal{B}_{j,v,\omega} \in {}^{s}\hat{\mathcal{B}}_{j,v}$: ${}^{s}f_{j,v,\gamma}$, ${}^{s}f^{\tau}_{j,v,\xi}$, ${}^{s}f^{\epsilon}_{j,v,\beta}$
Canonical decomposition	for each ${}^{s}f_{j,v,\gamma}$: ${}^{s}\hat{f}_{j,v,\gamma}$
Data availability	for each ${}^{s}f_{j,v,\gamma,\psi} \in {}^{s}\hat{f}_{j,v,\gamma}$: ${}^{s}\hat{f}_{j,v,\gamma,\psi}$
Send arrangement	for each ${}^{s}\mathcal{B}_{j,v,\omega} \in {}^{s}\hat{\mathcal{B}}_{j,v}$: sending_order
Send mode	for each ${}^{s}_{y}s_{j,v,v'} \in {}_{y}\hat{s}_{j,v}$ while executing ${}^{s}\mathcal{B}_{j,v,\omega} \in {}^{s}\hat{\mathcal{B}}_{j,v}$: timeout_1
Receive arrangement	for each ${}^{s}\mathcal{B}_{j,v,\omega} \in {}^{s}\hat{\mathcal{B}}_{j,v}$: receiving_order
Receive mode	for each ${}^{s}_{x}s_{j,v,v'} \in {}_{x}\hat{s}_{j,v}$ while executing ${}^{s}\mathcal{B}_{j,v,\omega} \in {}^{s}\hat{\mathcal{B}}_{j,v}$: timeout_2

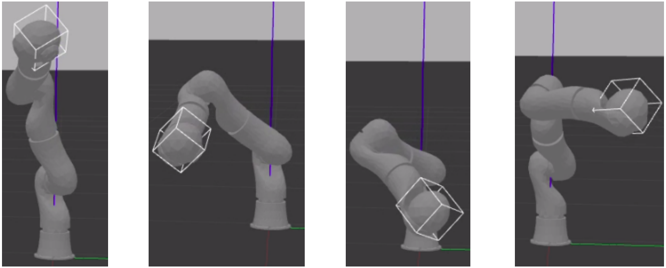
Translation



Experiments

- LWR4+ manipulator (specified from scratch),
- Table-tennis ball collecting robot (developed from scratch),
- Velma robot transferring balls (extended based on existing controller)

LWR4+ manipulator



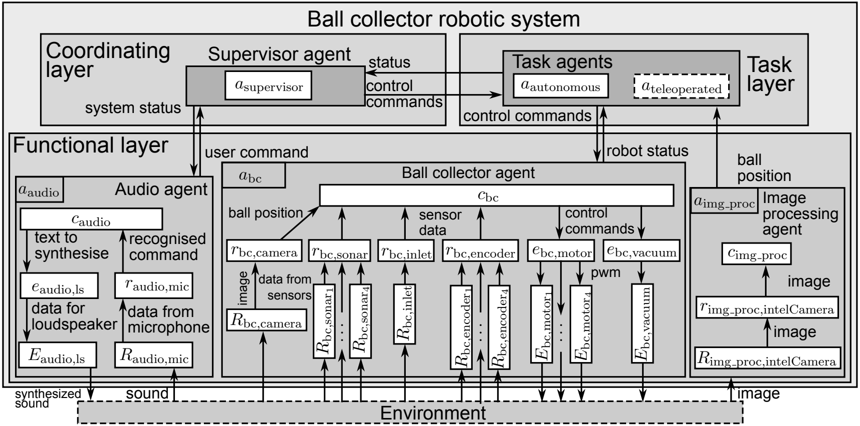
(a) (b) (c) (d)

Simulated LWR4+ manipulator with 7 degrees of freedom controlled using impedance control. Its end-effector moves along a circular trajectory (only its Cartesian position is controlled).

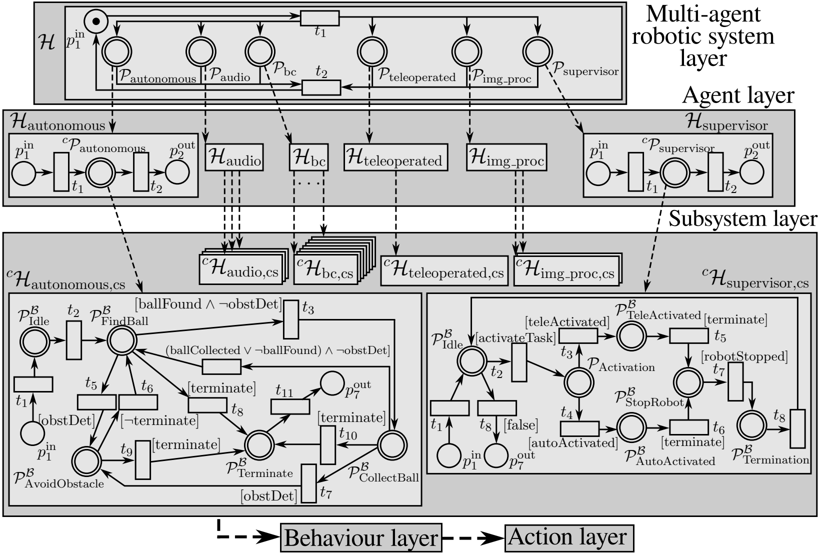
Table-tennis ball collecting robot

Ball collecting robot and the environment; camera images: (c) Raspberry Pi, (d) Intel Realsense D435; detected balls (in rectangles), closest ball (in circle).

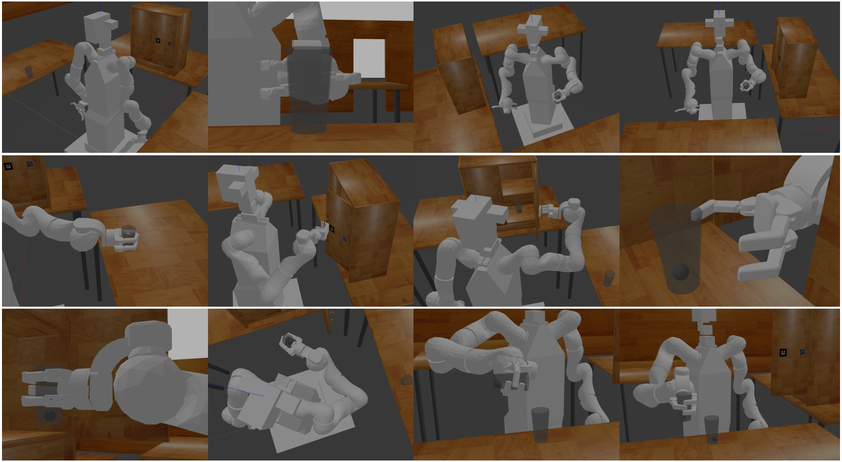
Ball-collecting robot – Structure



Ball-collecting robot – Activity

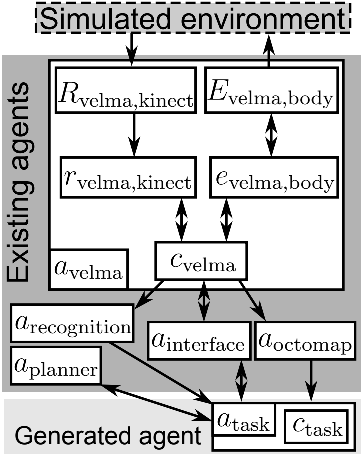


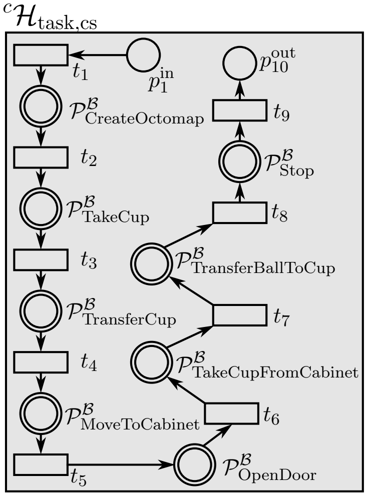
Velma robot transferring balls



Transfer of a ball from the cup located in the cabinet to the cup initially relocated from one table to the other.

Balls transfer – Structure and activity



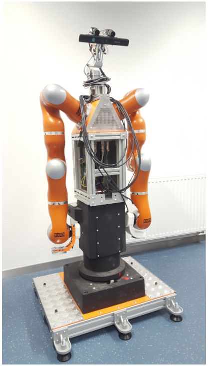


Conclusions

- RSSM methodology based on MDE has been proposed,
- a parameterized RSHPN meta-model was proposed,
- RSHPN Tool for RSHPN creation was developed,
- RSHPN network analysis was performed,
- RSSL domain language was developed for the specification of robotic systems,
- RSHPN meta-model and RSSL approaches were verified

Perspectives on continuing research

- RSHPN extension of the meta-model (e.g., time considerations),
- time-related analysis,
- generation of controller code for OROCOS,
- verification of the approach on a real Velma robot



Thank you for your attention!

The program is co-financed from the European Social Fund under the Operational Program Knowledge Education Development, a non-competition project entitled "International scholarship exchange of doctoral students and academic staff" implemented as part of the Activity specified in the application for co-financing of the project no. POWR.03.03.00-00-PN13 / 18.

ang. References I

- IFR - International Federation of Robotics, Executive Summary World Robotics 2020 Service Robots, [Online], Available: https://ifr.org/img/worldrobotics/Executive_Summary_WR_2020_ (2020).

- D. Kortenkamp, R. Simmons, D. Brugali, Robotic systems architectures and programming, in: B. Siciliano, O. Khatib (Eds.), Springer Handbook of Robotics, 2nd Edition, Springer, 2016, pp. 283–306. doi:10.1007/978-3-540-30301-5_9.

ang. References II

- M. Reichardt, T. Föhst, K. Berns, An overview on framework design for autonomous robots, it - Information Technology 57 (2) (2015) 75–84. doi:10.1515/itit-2014-1065.
- Robotics 2020 Multi-Annual Roadmap For Robotics in Europe - Horizon 2020 Call ICT-2017, SPARC The Partnership for Robotics in Europe: European Commision and euRobotics AISBL (2016). URL https://www.eu-robotics.net/cms/upload/topic_groups/H2020_Robotics_Multi-Annual_Roadmap_ICT-2017B.pdf

ang. References III

- M. Figat, C. Zieliński, Robotic system specification methodology based on hierarchical petri nets, IEEE Access 8 (2020) 71617–71627.
- M. Figat, C. Zieliński, Methodology of designing multi-agent robot control systems utilising Hierarchical Petri Nets, in: 2019 International Conference on Robotics and Automation (ICRA), 2019, pp. 3363–3369. doi:10.1109/ICRA.2019.8794201.
- M. Figat, C. Zieliński, "hierarchical petri net representation of robot systems", in: R. Szewczyk, C. Zieliński, M. Kaliczyńska (Eds.), Automation 2019, "Springer International Publishing", Cham, 2019, pp. 492–501. doi:10.1007/978-3-030-13273-6_46.

ang. References IV

M. Figat, C. Zieliński, R. Hexel, Fsm based specification of robot control system activities, in: 2017 11th International Workshop on Robot Motion and Control (RoMoCo), 2017, pp. 193–198. doi:10.1109/RoMoCo.2017.8003912.

C. Zieliński, M. Figat, R. Hexel, Communication within multi-fsm based robotic systems, Journal of Intelligent & Robotic Systems 93 (3) (2019) 787–805. doi:10.1007/s10846-018-0869-6.

ang. References V

C. Zieliński, M. Figat, R. Hexel, Robotic systems implementation based on fsms, in: R. Szewczyk, C. Zieliński, M. Kaliczyńska (Eds.), Automation 2018: Advances in Automation, Robotics and Measurement Techniques, Springer, 2018, pp. 441–452. doi:10.1007/978-3-319-77179-3_41.