EuRoSurge Workflow: From ontology to surgical task execution

E. De Momi, R. Perrone
Politecnico di Milano,
Milano Italy
Email: elena.demomi@polimi.it
roberta.perrone@mail.polimi.it

L. Schreiter, J. Raczkowsky
Karlsruhe Institute of Technology
Karlsruhe, Germany
Email: luzie.schreiter@kit.edu
joerg.raczkowsky@kit.edu

F. Boriero, M. Capiluppi, P. Fiorini
Università di Verona
Verona, Italy
Email: fabrizio.boriero@univr.it
marta.capiluppi@univr.it
paolo.fiorini@univr.it

***Abstract*—We present a workflow for design, deployment and verification of surgical robotic tasks. The workflow starts from the population of ontologies at two levels: task description and components specification. The ontologies are based on the technical and medical knowledge of the surgical task to be performed with a robot. From the ontologies, it is possible to design a model of the components and a finite state machine that represents the evolution of the task. This machine is verified using formal tools to avoid risks and preserve safety. Moreover a benchmarking phase assures the reliability of the components interaction. This workflow is applied to a simple case study.**

I. Introduction

The objective of the Coordination and Support Action *European Robotic Surgery* (EuRoSurge, FP7-ICT-2011-7) is to identify a methodology that is suitable for the integration of the research and manufacturing competences available in Europe in the two key areas of robotics and cognitive sciences, leading to easier product development and integration, standard settings, simpler technology transfer, and clearer identification of non-technical roadblocks. To this end, EuRoSurge aims at producing a reference workflow for the design, deployment and verification of a surgical robotic system.

In the preliminary part of the workflow an ontological modeling is used as knowledge base repository. Ontology is, by definition, a specification of a conceptualization [1]. The subject of ontology is the study of the categories of things that exist or may exist in some domain knowledge. Thus, a rigorous and exhaustive description of a specific domain is inside the ontology, and with the term domain, an area of knowledge, like for example the healthcare field, is meant [2]. An ontology is made of classes, individuals that populate classes, and properties that can link two different classes or can extend the meaning of a class. In medicine, ontologies have found large use because of problems of interoperability due to different nationalities and extraction of surgeons. Moreover, it is important to have a description of the medical knowledge that is understandable and processable by both men and machines. Medical ontologies like GALEN [3], SNOMED CT [4] and UMLS [5] are used as support for the healthcare practice from doctors, nurses, auxiliaries and also for the patients themselves. Ontologies can be used to formalize the workflow of a surgical procedure.

In our workflow, we use two different ontologies, with two different levels of granularity:

- The first ontology, named *TaskOntology*, will describe the considered task. It contains concepts related to all the possible states, transitions and commands that the component can assume. It could also contain information about the level of risk associated to the use of a component and related safety issues.
- The second ontology, named *ComponentOntology*, describes the set of components that participate to the scenario. All the components will be described by a set of property like inputs, outputs and specifications.

To control the surgical robotic system, we use a component-based software architecture [7] [8]. This software engineering technique suits our goal because it allows, thanks to its characteristics, to change the topology of the connections between the various software modules that implement the required functionalities. Currently, the most popular component-based robotic architectures are two: ROS and OROCOS. ROS [9] is a project created at Stanford (CA, USA) and mainly maintained by Willow Garage, broadly used in robotics communities. OROCOS [10] is another well known and used framework developed by the University of Leuven (BELGIUM).

The main feature of the surgical robotic context is the *heterogeneous* and *distributed* nature of the devices composing the setup. Indeed, the devices used in a teleoperation task [11] are spacially distributed and need to exchange a stream of real-time data. OROCOS has been chosen because, besides allowing the execution and transmission of real-time data [12], it integrates a finite state machine in each component to control its behaviour. This characteristic is of basic importance because it allows the designer to exploit reusability and modularity, e.g by changing the connections between the components without stopping or recompiling the software modules.

In order to design the software components corresponding to *ComponentOntology*, we need to find a tool that translates the ontological description into a coded model described by component name, input/output ports and type of data processed. Since we use the OROCOS architecture, we considered the two following tools: BRIDE[13]/VARP [14] and the script

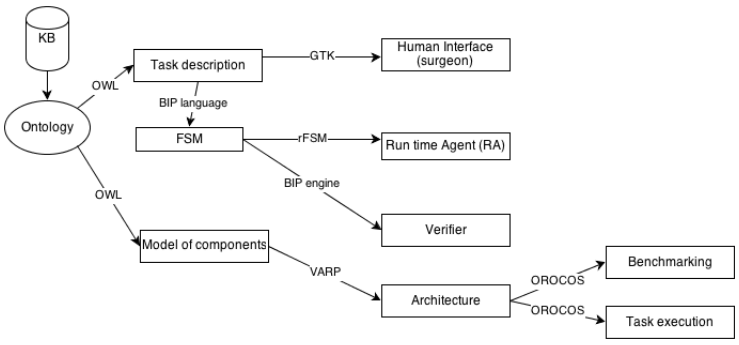

Fig. 1. The Eurosurge workflow. KB is the Knowledge Base repository, FSM is the Finite State Machine.

"orocreate-pkg" that is present in the OROCOS Toolchain. The idea is to start from a description ontology in OWL format [15], and automatically generate the code that could be integrated into the global architecture.

Once the components have been created and the functionalities implemented, it is necessary to coordinate them in order to follow the desired plan. For this purpose a deployer based on Lua [16] is used, an interpreted language used for the creation, the configuration and the implementation of the components.

In addition to the deployment and with the help of Lua, a finite state machine (FSM) can be used to describe and perform the operation task following a pre-defined plan. Indeed the FSM represents the evolution of the system to obtain the desired goal. It can also represent safety specifications and possible reconfiguration actions. A tool called RFSM described in [17] is used for designing such machine.

As for the components implementation, from the ontology description we can automatically generate a file readable by the RFSM toolbox. For its creation we used already existing ontologies exploiting available and reliable knowledge, like the suggested upper merged ontology (SUMO, [6]), that consists of general terms that can be used to connect and define our more specific terms. The final result is a running architecture generated by the two ontologies connected together: the $TaskOntology$ and the $ComponentOntology$. This is possible because concepts belonging to the two ontologies just explained can be linked, e.g concepts of the task ontology could have a relation with concepts of the component ontology.

Since the EuRoSurge project is interested in cognitive systems, the benchmarking phase afford tasks that cover the flexibility and autonomy of the aspired systems in order to respect the modularity. As explained above, one property of the reference architecture is the reusability: the components are created and should be used 'as they are', in different surgical-robotic contexts. For this reason a benchmark in the Eurosurge Project aims at comparing a similar component under the same conditions (task). The BRICS Project has launched three categories of metrics: cost, utility and reliability [18]. These metrics helped us in defining meaningful measurements based on a case study proposed in next section.

Formal methods tools, which help during the design phase to model synthesis of correct-by-construction systems, have been applied to high-level robot behaviours [20]. The aim of the verification phase is to verify the properties of modularity (two similar components with different characteristics but the same aim) and safety (ensuring that the system still performs in a region of safety) for the whole setup in a real-time environment. As we have figured out in [19] the technical aspect of safety is quite important in the area of robotic surgery. To respect the property of safety, the reliability metric of 'mission success' will represent in our case the accuracy property. To respect the modularity aspect, we are focusing on the BIP Framework. BIP is a component based framework for implementing embedded real-time systems in the area of robotics [21], [22] and [23]. We are especially interested in proving that our system is deadlock free, which means that in every reachable state a transition can still be activated. A derivative performance index of the first requirement is to prove reachability property. Hence, with the support of BIP, we are able to verify that the given finite state machine operates in a region of safety.

The above described workflow is presented in Fig. 1.

II. Case Study

We consider a manipulator moving freely in a room. The end effector of the manipulator has to reach a target, that might be a tissue in a surgical context. The position of the end effector in the task space is computed with a tracker, called Tracker 1. Since it is subject to faults, a second tracker (Tracker 2) is used as a backup. The setup is represented in Fig. 2, where a Staubli Puma 260 manipulator is used: it is controlled by a Galil 4080 control board and tracked using Naturalpoint Optitrack (Altair robotic laboratory setup) for Tracker 1 and ClaronTech Microtracker 2 for Tracker 2.

We use the *ComponentOntology* to describe particular instances, or classes, of components, thus the functional, physical and measurement aspects. In our scenario, we have four different components: *Robot, Supervisor, Tracker 1* and *Tracker 2*. Every component can have a label, some specific properties and ports. Ports can be input or output and, for every port, the type of data in input/output (e.g. integer, floating etc.) is defined, as shown in Fig. 3.

Fig. 2. Setup of the case study. T1 (Tracker 1) and T2 (Tracker 2) are represented by the circles: blue for T1 and red for T2.

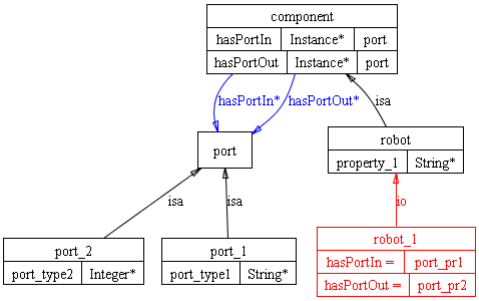

Fig. 3. Ontological description of a component. The component class is characterized by two properties that define the port of input and output. The component class is connected to the port class. In the figure the component class has an instance, which is a robot, that inherit all the properties of being a component.

The finite state machine (FSM) describing the operation plan is designed starting from the ontology that describes the surgical robotic task. For our case study, in the *TaskOntology* there are classes like: *State*, *Event* and *Transition*. The *State* class is populated with concepts like: *Init*, *Ready*, *DoTask*, *T1down* and *Goal*. The *Transition* class is populated with all the possible transitions that the robot can perform. Moreover, the *Transition* class will be characterized by two properties that states the starting state and the ending state of a specific transition.

Once the *TaskOntology* has described the finite state machine for our case study (Fig. 4), the verification phase implements it with BIP. A pseudo code of the finite state machine of Fig. 4 is given in Fig. 5: the events, states and transitions are described within commands **events, states, behaviour** respectively. The reader can see that the translation from the FSM to BIP code is straightforward. Here we added an event called e_μ representing the internal event moving the system from its initial state *Init* to the *Ready* state. This event is automatically generated by the FSM, but it has to be made explicit in the code. One of the most important problems to face is to check if the created FSM based on the ontology is deadlock free and if the states are in a valid order, via the reachability check. If and only if the verification is correct, the workflow can go on to the execution, which means that

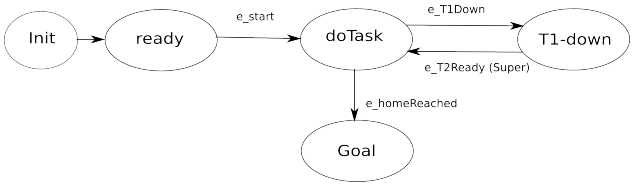

Fig. 4. Finite state machine of the case study

an automatic translation from the BIP language to RFSM is implemented and can be used for the execution of the task. We ran this check and verified that every state of the finite state machine in Fig. 4 is reachable using the BIP tools.

events e_μ(), e_start(), e_T1down(), e_T2ready(),
e_homeReached() , e_end()
states Init, Ready, DoTask, T1down, Goal
behaviour
initial to Init
on e_μ from Init to Ready
on e_start from Ready to DoTask
on e_homeReached from DoTask to Goal
on e_T1down from DoTask to T1down
on e_T2ready from T1down to doTask
on e_end from Goal to Goal

Fig. 5. BIP code of the case study.

Next step of the EuRoSurge workflow is the execution of the surgical task as shown in Fig. 6. In our simple case study, the robot performs a pre-planned trajectory. Tracker 1 calculates the position of the robot and sends it to the Supervisor component. If the error between the position calculated from the kinematics of the robot and the tracker measure is higher than a predefined threshold, the Supervisor will raise an event to the Run Time Agent (RTA). Notice that we are not interested in the algorithm used to choose the above mentioned threshold, since we are not discussing here the methods for fault detection. A simple study of the performance of Tracker 1 can lead to determine the right threshold to detect the fault

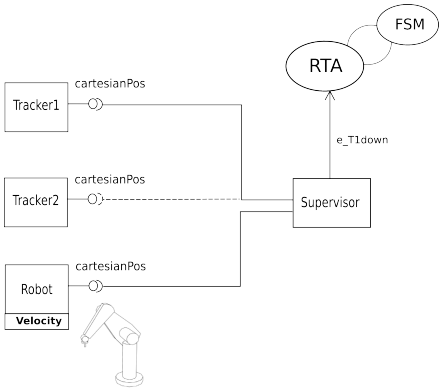

Fig. 6. Task execution of the case study: FSM is finite state machine, RTA is the Run-Time Agent.

and use residuals methods. In this case, the RTA will use this event to update the finite state machine that will return the new status of the surgical plan. The consequence is that the RTA will close the connection between Tracker 1 and the Supervisor and create a new connection between Tracker 2 and the Supervisor. As final result, in case of error, the second tracker automatically replaces the first one, allowing the continuation of the task. The RTA is the global coordinator that knows the described surgical task given by the FSM, controls the components, changes the architecture topology and verifies the desired properties using events exchanged with the supervisor.

The proposed benchmark starts by moving the robot with different estimated velocities from a starting point to an apriori defined target point, e.g. a point in the skin of a patient in case of a needle insertion operation. The benchmark takes place before the real surgical intervention will be done. The architecture provides a component, called *Reporter*, which is able to log all the required data for the benchmark. To estimate the accuracy of each tracker, the error between the given position of the robot and the given position of the trackers will be logged. The error value represents a comparative value for the accuracy of both tracking systems. The outcome of the benchmark phase depends strongly on the implementation and the environment. Hence the benchmark is performed with both trackers at the same time and the trajectory used is the same as in the surgical intervention scenario. Moreover, the collected data has been used in Matlab to calculate the error and to visualize the different error values depending on the defined velocities.

III. Conclusions

EuRoSurge workflow aims at giving guidelines for the design, implementation and verification of a surgical robotic system. In our workflow we started with an ontology describing both components and the surgical task, based on medical and technical knowledge. From this description, we model the components using a software (BRIDE/VARP, OROCOS) that translates the ontology written in OWL. The ongoing work aims at obtaining this translation automatically. The ontology provides the basis to design a finite state machine representing the evolution of the surgical task. It is necessary to involve the medical and technical experts to make this step. Indeed, the experts will help the designer to decide what parts of the ontology have to be used to design the specifications related to the specific instance of the surgical task. We are working on this cognitive aspect. Moreover, the verification procedure requires an automatic translation of the finite state machine from BIP to RFSM to design the run-time agent. Finally, a future extension of the workflow will include a human interface that should allow the surgeon to understand the level of risk it might take in performing specific actions. This interface will take into account the evolution of the system specified in the finite state machine and the task ontology. To this end, a level of risk has to be defined and an index needs to be agreed between the designer and the surgeon.

References

[1] T. Gruber, "What is an ontology," *Encyclopedia of Database Systems*, vol. 1, 2008.

[2] J. F. Sowa, "Ontology, metadata, and semiotics," in *Conceptual structures: Logical, linguistic, and computational issues*. Springer, 2000, pp. 55–81.

[3] A. L. Rector, J. E. Rogers, and P. Pole, "The galen high level ontology," *Studies in health technology and informatics*, pp. 174–178, 1996.

[4] L. Bos *et al.*, "Snomed-ct: The advanced terminology and coding system for ehealth," *Medical And Care Compunetics 3*, vol. 121, p. 279, 2006.

[5] O. Bodenreider, "The unified medical language system (umls): integrating biomedical terminology," *Nucleic acids research*, vol. 32, no. suppl 1, pp. D267–D270, 2004.

[6] I. Niles and A. Pease, "Towards a standard upper ontology," in *Proceedings of the international conference on Formal Ontology in Information Systems-Volume 2001*. ACM, 2001, pp. 2–9.

[7] D. Brugali and P. Scandurra, "Component-based robotic engineering (part i)[tutorial]," *Robotics & Automation Magazine, IEEE*, vol. 16, no. 4, pp. 84–96, 2009.

[8] D. Brugali and A. Shakhimardanov, "Component-based robotic engineering (part ii)," *Robotics & Automation Magazine, IEEE*, vol. 17, no. 1, pp. 100–112, 2010.

[9] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2, 2009.

[10] H. Bruyninckx, "Open robot control software: the orocos project," in *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, vol. 3. IEEE, 2001, pp. 2523–2528.

[11] B. Hannaford and P. Fiorini, "A detailed model of bi-lateral teleoperation," in *Proceedings of the IEEE international conference on systems, man and cybernetics*, vol. 1, 1988, pp. 117–121.

[12] H. Bruyninckx, P. Soetens, and B. Koninckx, "The real-time motion control core of the orocos project," in *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, vol. 2. IEEE, 2003, pp. 2766–2771.

[13] H. Bruyninckx, M. Klotzbücher, N. Hochgeschwender, G. Kraetzschmar, L. Gherardi, and D. Brugali, "The brics component model: a model-based development paradigm for complex robotics software systems," in *Proceedings of the 28th Annual ACM Symposium on Applied Computing*. ACM, 2013, pp. 1758–1764.

[14] L. Gherardi, "Variability modeling and resolution in component-based robotics systems," Ph.D. dissertation, Katholieke Universiteit Leuven, Belgium, 2013.

[15] D. L. McGuinness, F. Van Harmelen *et al.*, "Owl web ontology language overview," *W3C recommendation*, vol. 10, no. 2004-03, p. 10, 2004.

[16] M. Klotzbücher, P. Soetens, and H. Bruyninckx, "Orocos rtt-lua: an execution environment for building real-time robotic domain specific languages," in *International Workshop on Dynamic languages for RObotic and Sensors*, 2010, pp. 284–289.

[17] M. Klotzbuecher and H. Bruyninckx, "Hard real-time control and coordination of robot tasks using lua," in *Proceedings of the Thirteenth Real-Time Linux Workshop*, 2011, pp. 37–43.

[18] W. Nowak, Z. Zakharov, S. Blumenthal, and P. E., "Delivarable 3.1: Benchmarking for mobile manipulation and robust obstacle avoidance and navigation," 2010.

[19] M. Capiluppi, L. Schreiter, P. Fiorini, J. Raczkowsky, and H. Woern, "A fault analysis procedure for surgical robotic systems," in *Hamlyn Symposium on Medical Robotics 2013*, 2013.

[20] ——, "Modeling and verification of a robotic surgical system using hybrid input/output automata," in *press European Control Conference, Zurich*, 2013.

[21] T. Abdellatif, S. Bensalem, J. Combaz, L. De Silva, and F. Ingrand, "Rigorous design of robot software: A formal component-based approach," *Robotics and Autonomous Systems*, 2012.

[22] S. Bensalem, L. de Silva, F. Ingrand, and R. Yan, "A verifiable and correct-by-construction controller for robot functional levels," *Journal of Software Engineering for Robotics*, vol. 1, no. 2, pp. 1–19, 2011.

[23] A. Basu, M. Bozga, and J. Sifakis, "Modeling heterogeneous real-time components in bip," in *Software Engineering and Formal Methods, 2006. SEFM 2006. Fourth IEEE International Conference on*. Ieee, 2006, pp. 3–12.