Multi-Paradigm Design Space Exploration for Advanced Cyber-Physical Systems

Yon Vanommeslaeghe[a,b], Joachim Denil[a,b] and Paul De Meulenaere[a,b]

[a] CoSys-Lab (FTI), University of Antwerp, Belgium
[b] AnSyMo/CoSys, Flanders Make, Belgium

Abstract

As cyber-physical systems become more complex, the engineering domains involved in their design become more intertwined. In these situations, an integrated design approach is needed, where parameters in the different domains must be considered and evaluated together to ensure an efficient deployment. However, in such an approach, the design space expands dramatically. As such, methods that allow engineers from different domains to reason about cross-domain relationships are needed to derive an efficient design-space exploration (DSE) strategy. The goal of our research is to enable the concurrent design and optimization of the embedded control of the next-generation cyber-physical systems. To achieve this, we make use of ontologies to explicitly model and capture domain knowledge. By reasoning about the problem as a whole, using these ontologies, we can derive an efficient design space exploration process where we make use of different DSE techniques to solve smaller sub-problems over the different domains.

Keywords

Embedded Systems, Cyber-Physical Systems, Co-Design, Design Space Exploration, Ontologies

1. Introduction

The design of cyber-physical systems (CPS) is inherently a co-design process involving multiple engineering domains. Over the years the complexity of CPS, and the control and monitoring algorithms used therein have increased significantly. This trend is expected to continue with more advanced algorithms that contain models of the physics of the system they are meant to monitor and control. As these models are computationally expensive, embedded deployment for use in real-world applications is challenging, especially when real-time performance is required.

As these systems become more complex, the different domains involved (i.e. control and embedded) become intertwined, with decisions made in one domain imposing additional requirements or restrictions on another. For example, an autonomous vehicle may use model predictive control to determine which control inputs allow it to follow its intended trajectory as closely as possible. In this case, a larger prediction horizon may lead to better control performance, but also increases computational requirements. This complicates embedded deployment,

SEDES'2020: Software Engineering Doctoral Symposium, September 08, 2020, Online Conference

EMAIL: yon.vanommeslaeghe@uantwerpen.be (Y. Vanommeslaeghe); joachim.denil@uantwerpen.be (J. Denil); paul.demeulenaere@uantwerpen.be (P. De Meulenaere)
ORCID: 0000-0001-7871-2145 (Y. Vanommeslaeghe); 0000-0002-4926-6737 (J. Denil); 0000-0002-3706-6164 (P. De Meulenaere)

potentially making it infeasible. Similarly, the way in which the algorithm is deployed on the embedded platform can affect the control performance. For example, unexpected delays due to the temporal behaviour of the embedded platform can degrade control performance.

In these situations, a traditional, sequential design process becomes insufficient to ensure efficient deployment and integration. As such, it is necessary to regard this as a multi-domain co-design process, where parameters in both the control and embedded domain must be considered and evaluated to ensure the quality of the software and hardware configuration after deployment. However, in such an integrated approach, the combined design space expands dramatically due to the increased number of degrees of freedom. Therefore, the design space exploration (DSE) needs to be performed efficiently. We propose the use of ontologies [1] to facilitate the optimization of systems involving multiple engineering domains, by capturing domain knowledge and using it to guide the DSE process. In this guided design space exploration process, different DSE techniques are used to solve different parts of the problem across the different domains. As such, we refer to this as multi-paradigm design space exploration.

2. State of the Art

A lot of research has already been done regarding hardware/software co-design. Frameworks such as Metropolis [2] and Metro II [3], generally follow the principle of the platform-based design [4] methodology: the "orthogonalization of concerns", i.e. separation of different aspects of design. For complex cyber-physical systems this is no longer realistic, as the different domains become more intertwined. More recently, efforts have been made regarding the co-optimization of CPS designs in multiple domains and on resource-aware embedded control system design [5].

Aminifar et al. [6] address the controller-server co-design problem for multiple control tasks running on a shared platform. Here, virtual servers are used to divide the computational resources among the different tasks. Their goal is to optimize the server configurations, together with the sample periods of the controllers to ensure their stability. To achieve this, they first compute a stability curve for each controller. This is then used to determine the optimal configuration of server parameters and controller periods to minimize resource utilization while ensuring system stability. Similarly, Roy et al. [7] consider the design of a (distributed) FlexRay-based embedded control system. Here, they co-optimize the controller design together with the FlexRay schedule, resulting in a Pareto front depicting the trade-off between control performance and bus-utilization.

From the state of the art, it is clear that there is a benefit to co-optimization between the control- and embedded domains and that it is possible to come up with efficient strategies for design-space exploration and optimization. However, it is apparent that determining these strategies is not straightforward. It generally requires a good grasp of both the control- and embedded domain and their interdependencies (domain knowledge). As such, methods that allow engineers from different domains to reason about these cross-domain relationships are needed for them to derive an efficient design-space exploration strategy. It is here that ontologies can be used to explicitly capture the combined domain knowledge and leverage it to guide the DSE process.

Previously, Vanherpen et al. [8] worked on combining contract-based design [9] with ontological reasoning to reason about the content of such contracts. Similarly, Dávid et al. [10] use ontologies in the context of (in)consistency management. However, neither explored the use of ontologies in determining an efficient DSE process.

3. Research Objectives

The ultimate goal of this research is to enable multidomain co-design for the deployment of novel control and monitoring algorithms on embedded platforms. As such, the *main objective* of this research is to define the methods, tools, and techniques for the concurrent design and concurrent optimization of the embedded control of next-generation CPS by means of multi-paradigm design space exploration and domain knowledge. This can be further decomposed into the following *sub-objectives*:

Sub-objective 1: Definition of a method to capture and refine the common domain knowledge. This method will need to be able to elicit the domain knowledge from the control and the embedded engineer. It will also need to support the semi-automatic characterization of case-specific parameters such as execution times, memory usage, etc.

Sub-objective 2: Definition of a design space exploration technique for fast exploration of design options early on in the design process. The combination of the domain knowledge and early design models based on virtual hardware must steer the engineer towards a quick evaluation of the effect of different candidate high-level design choices, such as processor type, architecture, etc.

Sub-objective 3: Definition of a technique to optimize the DSE process for co-design problems, with explicit use of the captured domain knowledge. The design space for the deployment of the next generation CPS is growing huge due to the increased number of both control and embedded design choices. The goal is to optimize the DSE process by using the relationships between the parameters and other information registered in the knowledge base.

4. Past Work and Preliminary Results

Our earlier work focused on the co-simulation of embedded platform models with models of the application expected to run thereon [11]. Here, we considered the embedded deployment of virtual sensing algorithms, which include models of the physics. In this work, we explicitly model the interfaces and scheduler of the embedded system, and combine this with the original application model as illustrated in Figure 1. By doing this, it becomes possible to simulate the impact of certain hardware aspects of the embedded platform, such as additive measurement noise and timing behaviour, on the application. As this allows the temporal and functional behaviour of the application to be verified on the model level, it allows engineers to quickly determine the impact of different design decisions on application performance. This is demonstrated using the embedded realisation of a virtual sensor, where we evaluate the impact of

ADC resolution and potential deadline misses on the accuracy of the virtual sensor. As such, this work serves *sub-objective 2*, as it allows for the construction of virtual prototypes to quickly evaluate the impact of high-level design decisions.

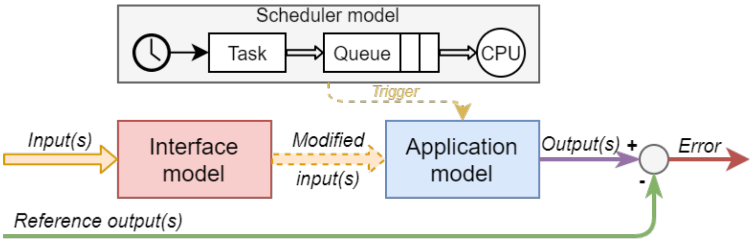


Figure 1: Setup for simulating the impact of embedded platform effects on applications.

More recent work of ours focused on using domain knowledge, captured in an ontology, to guide the design space exploration process for an advanced CPS, such as an advanced control algorithm for brushless DC (BLDC) motors [12]. In this work, we show how domain knowledge can be used to solve a multi-domain optimization problem. First, we build an ontology of important design parameters and their interdependencies. This is illustrated in Figure 2, which shows the ontology for the advanced BLDC control algorithm. In this figure, decision variables and objectives are indicated in blue and green respectively, and arrows indicate which properties affect which directly. Additionally, the plus and minus signs indicate how these parameters affect each other. Building such an ontology allows engineers from different domains to reason about the design space exploration process. By doing this, it becomes clear how information or evaluations in one domain can be used to constrain parameters in an other and where trade offs can be made. This allows us to determine an efficient design space exploration process to quickly narrow down the design space to a subset of likely good candidates. For the case of the BLDC control algorithm, the design space is quickly limited to 3% of its original size, without discarding top candidates regarding system performance. This work mostly contributes towards *sub-objectives 1* and *3*.

Recently, we have extended this method by introducing characterization and ontology updat-

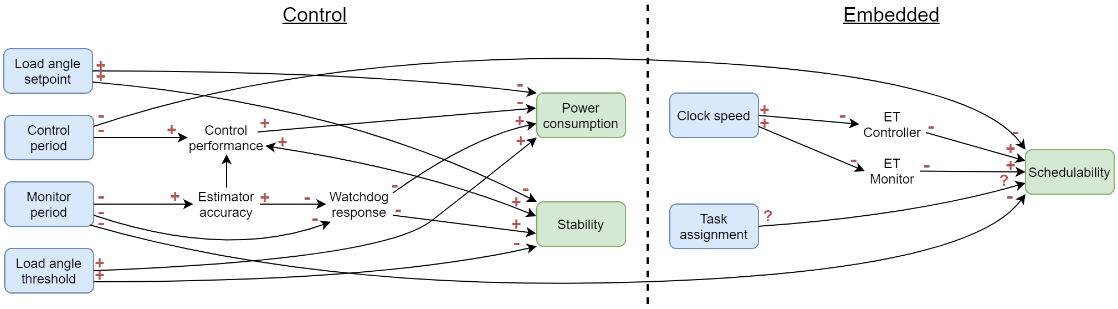


Figure 2: Ontology of the BLDC controller use case.

ing steps [13]. These steps allow for the systematic updating of the ontology as new information or insights become available during the design process. Additionally, we add sensitivity information which allows us to prioritize certain (more sensitive) parameters during the design space exploration process. This further contributes towards *sub-objectives 1*, *2* and *3*.

5. Future Work and Expected Results

The results shown in previous work are promising, especially when keeping in mind the development of new, advanced cyber-physical systems. In future work, we intend to extend this work in two main ways:

Modelling of Domain Knowledge Currently, the ontologies we use are built up manually and only exists as a drawing. As previously mentioned, Vanherpen et al. [8] worked to combine contract based design (CBD) [14] with ontological reasoning. This resulted in an ontology specific to the co-design of cyber-physical systems. We will investigate how this existing ontology can be extended specifically to support our approach. Additionally, we will investigate which parts of the domain knowledge can be captured in a generic way and which parts are application specific. This work will contribute further to *sub-objective 1*.

Automation Currently, our method requires manual work to build and update the ontology, and to derive the design-space exploration process. We will investigate to what extent this process can be derived automatically from the captured domain knowledge, for example by detecting certain patterns in the ontology. This will contribute further to *sub-objectives 2* and *3*. We also intend to investigate if we can use the ontology itself to determine which relationships need to be further characterized. For example by running sensitivity analyses or other experiments. This would further contribute towards *sub-objective 1*.

Additionally, in previous work we made use of simple case studies, consisting of one algorithm deployed on a single embedded platform. To further develop and validate our methods, we will make use of a more complex use case. This use case should consist of multiple separate algorithms, possibly with conflicting interests, running on a shared embedded platform, where they are competing for resources. As such, we will need to optimize the configurations of the different algorithms together with the resource allocation to ensure correct overall system behaviour. This case will also be used to benchmark our methods and compare them to the state of the art. Overall, we are aiming to reduce the computation time for the design space exploration by a factor of 10, while still finding the best or similarly performing solutions.

6. Conclusions

In previous work we have shown that we can use domain knowledge, captured in an ontology, to derive an efficient design space exploration process. This was demonstrated on an academic use case where we were able to quickly reduce the design space to 3% of its original size, without discarding top candidates. In future work we will further develop our method, along with supporting tools to model the domain knowledge and to derive an efficient design space process from this. In this design space exploration process, different DSE techniques will be used to solve different parts of the problem. As such, we refer to this as multi-paradigm design space exploration.

References

[1] T. R. Gruber, et al., A translation approach to portable ontology specifications, Knowledge acquisition 5 (1993) 199–221.

[2] F. Balarin, Y. Watanabe, H. Hsieh, L. Lavagno, C. Passerone, A. Sangiovanni-Vincentelli, Metropolis: An integrated electronic system design environment, Computer 36 (2003) 45–52.

[3] A. Davare, D. Densmore, L. Guo, R. Passerone, A. L. Sangiovanni-Vincentelli, A. Simalatsar, Q. Zhu, Metro ii: A design environment for cyber-physical systems, ACM Transactions on Embedded Computing Systems (TECS) 12 (2013) 49.

[4] A. Sangiovanni-Vincentelli, G. Martin, Platform-based design and software design methodology for embedded systems, IEEE Design & Test of Computers 18 (2001) 23–33.

[5] W. Chang, S. Chakraborty, et al., Resource-aware automotive control systems design: A cyber-physical systems approach, Foundations and Trends® in Electronic Design Automation 10 (2016) 249–369.

[6] A. Aminifar, E. Bini, P. Eles, Z. Peng, Bandwidth-efficient controller: server co-design with stability guarantees, in: Proceedings of the conference on Design, Automation & Test in Europe, European Design and Automation Association, 2014, p. 55.

[7] D. Roy, L. Zhang, W. Chang, S. Chakraborty, Automated synthesis of cyber-physical systems from joint controller/architecture specifications, in: 2016 Forum on Specification and Design Languages (FDL), IEEE, 2016, pp. 1–8.

[8] K. Vanherpen, J. Denil, I. Dávid, P. De Meulenaere, P. J. Mosterman, M. Torngren, A. Qamar, H. Vangheluwe, Ontological reasoning for consistency in the design of cyber-physical systems, in: 2016 1st International Workshop on Cyber-Physical Production Systems (CPPS), IEEE, 2016, pp. 1–8.

[9] A. Benveniste, B. Caillaud, D. Nickovic, R. Passerone, J.-B. Raclet, P. Reinkemeier, A. Sangiovanni-Vincentelli, W. Damm, T. A. Henzinger, K. G. Larsen, et al., Contracts for system design, Foundations and Trends® in Electronic Design Automation 12 (2018) 124–400.

[10] I. Dávid, J. Denil, K. Gadeyne, H. Vangheluwe, Engineering process transformation to manage (in) consistency, in: Proceedings of the 1st International Workshop on Collaborative Modelling in MDE (COMMitMDE 2016) co-located with ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems (MoDELS 2016) St. Malo, France, October 4, 2016/Muccini, Henry [edit.]; et al., 2016, pp. 7–16.

[11] Y. Vanommeslaeghe, P. De Meulenaere, J. Denil, F. Cosco, B. Forrier, J. Croes, Towards co-simulation of embedded platforms and physics-based models, in: 2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), IEEE, 2018, pp. 97–100.

[12] Y. Vanommeslaeghe, J. Denil, J. De Viaene, D. Ceulemans, S. Derammelaere, P. De Meulenaere, Leveraging domain knowledge for the efficient design-space exploration of advanced cyber-physical systems, in: 2019 22nd Euromicro Conference on Digital System Design (DSD), IEEE, 2019, pp. 351–358.

[13] Y. Vanommeslaeghe, J. Denil, J. De Viaene, D. Ceulemans, S. Derammelaere, P. De Meulenaere, Ontological reasoning in the design-space exploration of advanced cyber-physical systems, 2020. Manuscript submitted for publication.

[14] A. Benveniste, B. Caillaud, D. Nickovic, R. Passerone, J.-B. Raclet, P. Reinkemeier, A. Sangiovanni-Vincentelli, W. Damm, T. Henzinger, K. G. Larsen, Contracts for system design, Ph.D. thesis, Inria, 2012.

A. Work Plan

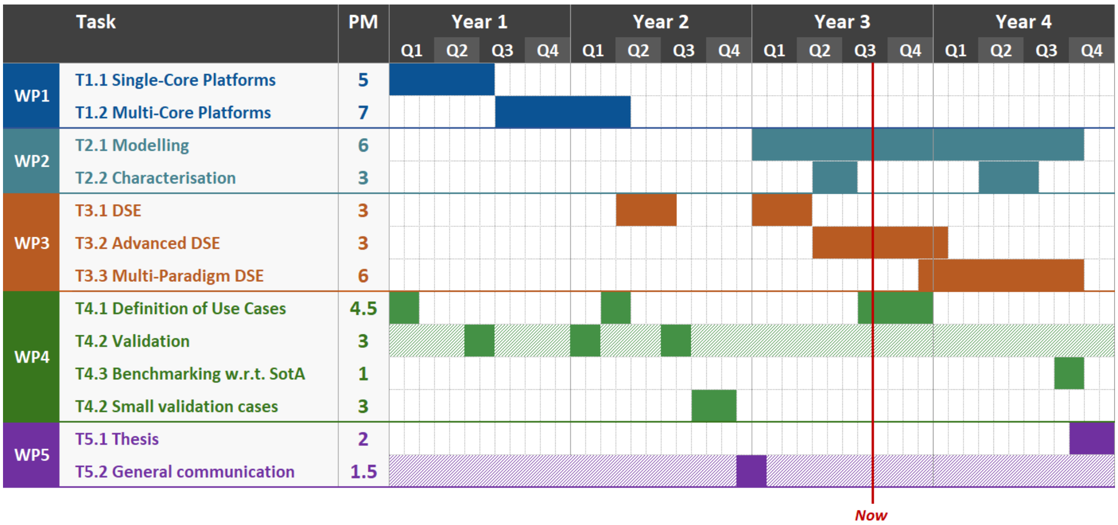


Figure 3: Timeline of the research.

The overall timeline of the research is shown is shown in Figure 3. On this chart, certain tasks are overlapping to allow input between the two tasks. This is particularly important for T2.1 (Modelling), as there is a lot of bilateral information exchange between this and the tasks of WP3 due to the nature of this research. Additionally, part of the time allotted to T4.2 and 5.2 is for general purpose validation and dissemination during this research. This is indicated with an overall hatched pattern, whereas notable event are marked with filled blocks. Lastly, the red line indicates the current point in the plan at the time of writing. The work packages themselves are described in detail in the following subsection.

A.1. Work Packages

WP1: Virtual hardware (12 months total)

This work package concerns the modelling of embedded platforms. Here, I will look at different formalisms to model the relevant hardware characteristics. I will also look at co-simulation techniques, such as FMI, to simulate the hardware models together with (a model of) the software. After this, I will investigate how closely the simulated hardware approximates the real hardware. This will support the DSE process by enabling the generation of virtual prototypes.

Task 1.1: Single-core embedded platforms (5 months)

In this task, I will investigate which formalisms and co-simulation techniques are appropriate to model single-core embedded platforms. This includes things like the scheduler, but also peripherals such as the ADC.
Dependencies: None

Task 1.2: Multi-core embedded platforms (7 months)

In this task, I will investigate which formalisms and co-simulation techniques are appropriate to model multi-core embedded platforms. Here, I will focus on mechanisms specific to multi-core embedded platforms, such as shared memory.
Dependencies: Task 1.1

Deliverables

- D1.1: Report on the modelling and simulation of single-core embedded platforms.
- D1.2: Report on the modelling and simulation of multi-core embedded platforms.

WP2: Domain knowledge (9 months total)

This work package mainly deals with the gathering and modeling of the domain knowledge. This needs to support the DSE process.

Task 2.1: Modelling domain knowledge (6 months)

I will investigate how the domain knowledge can be expressed in the most appropriate languages so they enable the fast exploration during the contract negotiation phase (*sub-objective 2*) and the multi-paradigm design space exploration during the optimisation phase (*sub-objective 3*). Currently, I consider two possibilities:

- **Ontologies** can be used to capture the relationships between different parameters in both domains; e.g., update frequency in the control domain relates to the task period in the embedded domain.

- **Constraints** can be used to express heuristics (e.g., large matrix multiplications can be performed more efficiently on a GPU) and to express certain requirements (e.g. safety critical parts need to be mapped to a platform which is ASIL-D certified).

I will also look at extracting relevant parameters that can be used in these heuristics, e.g. determining memory usage using static code analysis tools such as StackAnalyzer. Heuristics might be best modelled using a pattern catalogue and later operationalised as constraints in a constraint language.
Dependencies: Task 2.2, Task 3.3

Task 2.2: Automatic characterisation (3 months)

In this task, a framework will be built to allow the (semi-)automatic characterisation of applications on different types of hardware. This will allow me to automatically run experiments, effectively front-loading the characterization that is normally only done later in the design phase. The application code will need to be instrumented, here existing tools such as RapiTime can be used. Appropriate languages need to be used that capture the information obtained so it can later be used by the DSE. This can be used to characterize relationships captured in the ontology.
Dependencies: Task 2.1

Deliverables

- D2.1: Report on the relevant domain knowledge. This includes a description of the relations, captured in an ontology, heuristics and/or constraints.

- D2.2: A Framework for the automatic characterization of software on different types of hardware.

WP3: Design space exploration (12 months total)

Task 3.1: Design space exploration for embedded deployment (3 months)

In this task, I will investigate how existing design space exploration techniques can be used to support the efficient embedded deployment of advanced virtual sensing algorithms. Here, I will mainly focus on determining parallelization options for advanced algorithms on multi-core embedded platforms.
Dependencies: Task 1.2

Task 3.2: Advanced design space exploration techniques (3 months)

In this task, I will investigate and catalogue which design space exploration techniques are most suited to solve different common sub-problems for the design of advanced cyber-physical systems. I will apply these techniques to the academic use case, which also serves as preparation for the benchmarking task.
Dependencies: Task 3.1, Task 4.1

Task 3.3: Multi-paradigm design space exploration (6 months)

This task deals with the development of the DSE method(s) specifically for this problem. This will be a combination of techniques that need to be applied at different levels of abstraction. I will also determine where the relations, heuristics, and gathered measurements can be used most effectively to guide the DSE process.
Dependencies: Task 2.1, Task 3.2

Deliverables

- D3.1: Report on DSE techniques for the efficient embedded deployment of advanced virtual sensing algorithms.
- D3.2: Catalogue of DSE techniques for the design of cyber-physical systems.
- D3.2: Report on multi-pardigm DSE methods and techniques specifically for the "co-design exploration" and "co-design optimization".

WP4: Validation and benchmarking (11.5 months total)

This work package deals with everything regarding validation and benchmarking, including the definition and development of use cases, test setups, etc.

Task 4.1: Definition of use cases (4.5pm)

This time is reserved for the definition and elaboration of use cases over the course of the project.
Dependencies: None

Task 4.2: Validation of developed techniques (3 months)

This time is reserved for the intermediate validation of developed techniques over the course of this project.
Dependencies: None

Task 4.3: Benchmarking with respect to the state-of-the-art (1 months)

In this task, the developed methods and tools will be benchmarked. Here, I will compare the results of the developed multi-paradigm design space exploration approach to the results obtained using a conventional approach using the methods from Task 3.2. Results of the two approaches will be compared; e.g., by checking how many of the top 5 designs were also identified by the guided DSE in comparison to the amount of time saved. As such, it serves as the final validation of the developed techniques.
Dependencies: Task 3.2

Task 4.4: Small validation cases (3 month)

Together with companies two small validation cases (SVC) of 1,5 person months each will be defined. Here, the developed tools and techniques will be applied to a small case provided by the companies. These SVC will be used to further validate the developed tools and techniques, but also to motivate these companies to take part in defining a follow-up trajectory.
Dependencies: None

Deliverables

- D4.1: Definition of use cases with their required models, documentation, requirements, etc.
- D4.3: Report of the effectiveness of the developed methods and tools, together with deployment options for the specific case used.
- D4.4: For each small validation case there will be a report for the company.

WP5: Dissemination and valorisation (3.5 months total)

Task 5.1: Writing thesis (2 months)

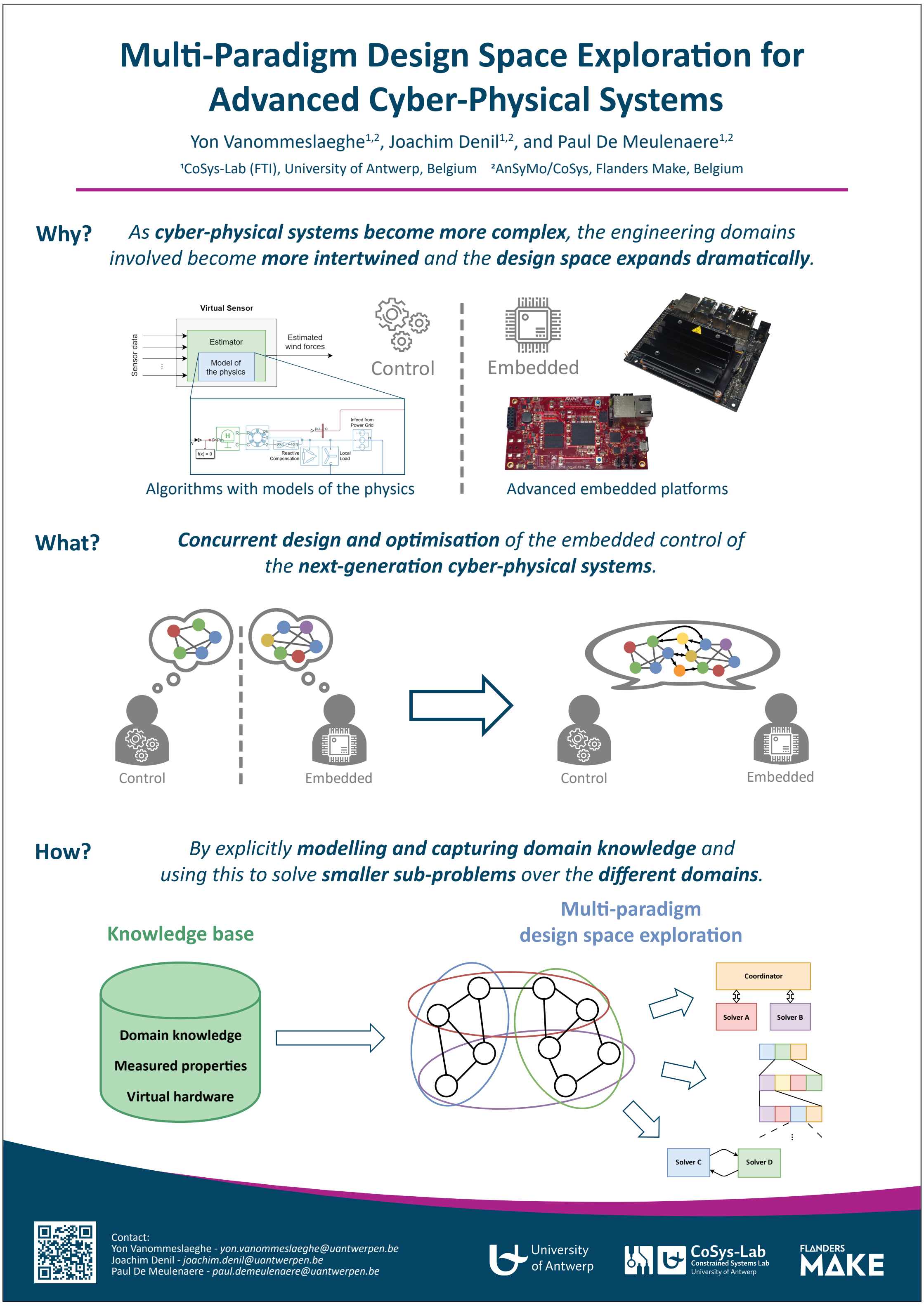
This time is reserved for writing the thesis.

Task 5.2: General communication (1.5 months)

This time is reserved for anything to do with general communication. For example, building a demonstrator, making a video or infographic, attending conferences, open campus days, etc.

B. Poster

A poster presenting an overview of this research is shown in Figure 4.



Multi-Paradigm Design Space Exploration for Advanced Cyber-Physical Systems

Yon Vanommeslaeghe[1,2], Joachim Denil[1,2], and Paul De Meulenaere[1,2]

[1]CoSys-Lab (FTI), University of Antwerp, Belgium [2]AnSyMo/CoSys, Flanders Make, Belgium

Why?

*As **cyber-physical systems become more complex**, the engineering domains involved become **more intertwined** and the **design space expands dramatically**.*

Algorithms with models of the physics

Advanced embedded platforms

What?

***Concurrent design and optimisation** of the embedded control of the **next-generation cyber-physical systems**.*

How?

*By explicitly **modelling and capturing domain knowledge** and using this to solve **smaller sub-problems** over the **different domains**.*

Knowledge base

Domain knowledge

Measured properties

Virtual hardware

Multi-paradigm design space exploration

Contact:
Yon Vanommeslaeghe - *yon.vanommeslaeghe@uantwerpen.be*
Joachim Denil - *joachim.denil@uantwerpen.be*
Paul De Meulenaere - *paul.demeulenaere@uantwerpen.be*

Figure 4: The poster with an overview of this research.