TEACHING: A computing Toolkit for building Efficient Autonomous appliCations leveraging Humanistic INtelliGence

Davide Bacciu
davide.bacciu@unipi.it
Department of Computer Science, University of Pisa
Pisa, Italy

Konstantinos Tserpes
tserpes@hua.gr
Department of Informatics and Telematics, Harokopio University
Athens, Greece

Massimo Coppola
massimo.coppola@isti.cnr.it
Institute of Information Science and Technologies, National Research Council of Italy
Pisa, Italy

Georg Macher
georg.macher@tugraz.at
Department of Informatics and Telematics, TU Graz
Graz, Austria

Claudio Gallicchio
claudio.gallicchio@unipi.it
Department of Computer Science, University of Pisa
Pisa, Italy

Omar Veledar
omar.veledar@avl.com
AVL
Graz, Austria

Anna Maria Anaxagorou
aanaxagorou@itml.gr
Information Technology for Market Leadership
Athens, Greece

Patrizio Dazzi
patrizio.dazzi@unipi.it
Department of Computer Science, University of Pisa
Pisa, Italy

ABSTRACT

TEACHING proposes a distributed, trustworthy AI integrating continuous human feedback, supporting CPSoS application design and deployment. TEACHING envisions an intelligent environment, empowering humans through cybernetic assistance. It advances autonomous safety-critical systems, improving safety, reliability and acceptability through human-centred design and formal validation crossing paradigms. TEACHING brings humans and AI together, enabling participatory development, optimisation and oversight.

CCS CONCEPTS

• **Hardware** → *Emerging architectures*; • **Computer systems organization** → *Embedded and cyber-physical systems*; • **Computing methodologies** → *Artificial intelligence*; • **Human-centered computing**;

KEYWORDS

humanistic intelligence; cyber-physical systems-of-systems; distributed artificial intelligence

ACM Reference Format:
Davide Bacciu, Konstantinos Tserpes, Massimo Coppola, Georg Macher, Claudio Gallicchio, Omar Veledar, Anna Maria Anaxagorou, and Patrizio Dazzi. 2023. TEACHING: A computing Toolkit for building Efficient Autonomous appliCations leveraging Humanistic INtelliGence. In *Proceedings of the 3rd Workshop on Flexible Resource and Application Management on the Edge (FRAME '23), June 20, 2023, Orlando, FL, USA*. ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3589010.3594886

FRAME '23, June 20, 2023, Orlando, FL, USA

ACM ISBN 979-8-4007-0164-1/23/06…$15.00
https://doi.org/10.1145/3589010.3594886

1 INTRODUCTION

Driven by the automation capabilities of Artificial Intelligence (AI), industry and society are experiencing the transformative impact of the autonomous systems revolution. Cyber-Physical Systems of Systems (CPSoS) define a diverse and dynamic environment where autonomy is fundamental to managing complex interactions between virtual and physical worlds with minimal human intervention [1]. However, even with the highest level of autonomy, the human is a variable that cannot be left out of the CPSoS equation. This is especially true in safety-critical scenarios such as autonomous transport. TEACHING [5] embraces a vision of the human at the centre of autonomous CPSoS by embracing the concept of humanistic intelligence [3], where the cybernetic and biological entities cooperate in mutual empowerment towards a common goal and where human feedback becomes a critical driver of CPSoS adaptivity. TEACHING addresses this challenge by integrating AI with fundamental safety and reliability concepts arising from AI-human-CPSoS interactions and considering their impact on the underlying computing system. TEACHING is developing a human-aware CPSoS for autonomous safety-critical applications based on a distributed, energy-efficient and reliable AI, using edge computing platforms that integrate a specialised computing fabric for AI and in silico support for intelligent cyber security. The aim is to develop a computing software and system that supports developing and deploying adaptive and reliable CPSoS applications. This will enable sustainable human feedback to control, optimise and personalise the services provided. TEACHING aims to impact the development of autonomous safety-critical systems, improving their safety, reliability and overall acceptability.

2 TEACHING PERSPECTIVE

Using specialised AI computing fabrics and in-silico support for intelligent cybersecurity solutions, TEACHING develops human-centric CPS for autonomous safety-critical applications based on distributed, energy-efficient and trustworthy AI. AI methods enable continuous human feedback to control, optimise and personalise services and support the design and deployment of autonomous, adaptive and reliable CPS applications. TEACHING envisages an intelligent environment where human and cybernetic entities work in synergy. The latter provides the former with convenient, tailored and reliable interaction driven by implicit human feedback through physiological responses to CPS manipulation.

3 TEACHING ARCHITECTURE

One conceptual view of the TEACHING Platform was presented in the project report "D1.1: Report on TEACHING related technologies SoA and derived CPSoS requirements" [6]. This conceptual architecture follows the rationale of layered architectures, where each layer offers services to the one above. Instantiations of the conceptual architecture may include implementations that merge layers, similar to ISO/OSI and TCP/IP. The starting point for designing the architecture of the TEACHING Platform has been the TEACHING objective which states "a computing platform and the associated software toolkit supporting the development and deployment of autonomous, adaptive and dependable CPSoS applications". As such, at the top layer, we place the CPSoS applications that are meant to be supported by the computing platform and the software toolkit, i.e. the TEACHING Platform. Based on our definition of CPSoS applications, i.e. the applications that meet a certain number of Non-functional requirements (NFRs), we provide a layer whose components meet those NFRs. This layer is meant to provide the specification of the software toolkit. The underlying layers form the TEACHING computing platform. They start with the layer meant to provide all the supporting software tools that will allow the development of the CPSoS applications and meet the functional requirements. The layer below is meant to specify how the computing platform will deal with interoperability issues, homogenising the underlying computing and network infrastructures. The final layer deals with the specification of the infrastructure. In what follows, we provide a more detailed view of the TEACHING Platform. The TEACHING platform comprises five layers, each providing services to the one above. At the bottom of the stack, we have the infrastructure layer.

3.1 Infrastructure Layer

The infrastructure layer comprises various heterogeneous infrastructures exposed through an embedded system OS and the cloud/edge resources. TEACHING assumes that access to the resources of those infrastructures is a priori possible. On that premise, the first task of TEACHING is to homogenise those resources, something that is the main functionality of the Infrastructure Abstraction Layer.

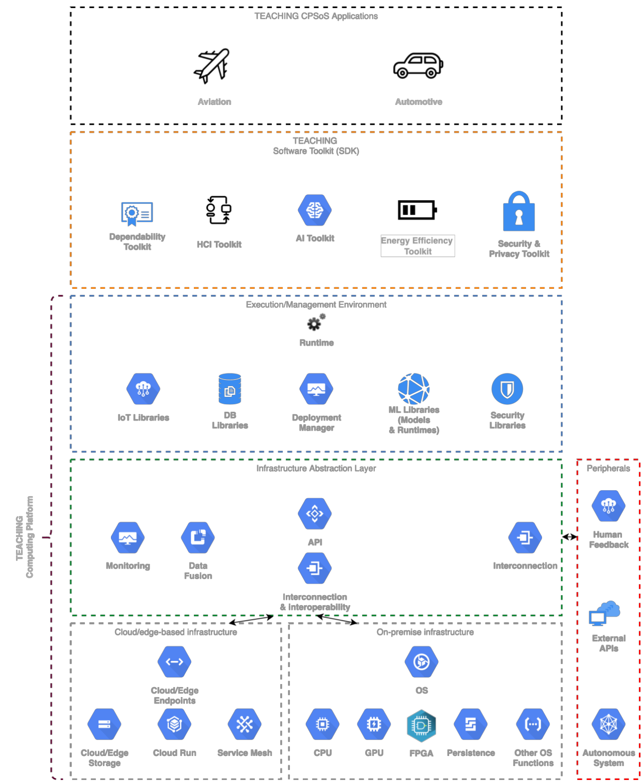

Figure 1: TEACHING architecture

3.2 Infrastructure Abstraction Layer (IAL)

The IAL provides a single abstraction layer for the execution of applications (code or components). Essentially it homogenises the underlying infrastructures providing a single API to deploy, execute and monitor resources and application components. This layer also caters for implementing I/Os, with the underlying persistence layers and the supported peripherals, i.e., the target autonomous system (CPS), external APIs (e.g., web services), but most importantly, with the mechanisms that provide the human feedback.

3.3 Execution/Management Environment (EME)

The EME exposes a single API that facilitates the execution and lifecycle management of the application components. It provides the runtime for that purpose and integrated libraries, implemented at a low-abstraction language, providing services and optimisations at the top layers. Such libraries include ML runtimes like those of TensorFlow [2] and PyTorch [4] or ML optimisations in Python, C++, Java, etc. It also includes libraries for managing IoT solutions (e.g., OS-IoT) implementing IoT protocols such as OneM2M. Other libraries include the DB and security libraries, ensuring this functionality is provided to the layers above.

3.4 TEACHING Software Toolkit (SDK)

The TEACHING SDK provides the framework to implement CPSoS applications. It provides APIs to implement applications that can run on the TEACHING platform using the best CPSoS services. The TEACHING SDK supports 6 toolkits:

3.4.1 AI toolkit. The TEACHING AI toolkit is the software library that allows the developer to invoke learning modules, set up training or inference procedures, etc. The AI toolkit has the appropriate wirings with the underlying layers to deploy and run the ML components at the appropriate resources (e.g., GPUs). It facilitates the I/Os and dataset management.

3.4.2 HCI toolkit. TEACHING HCI toolkit allows the software developer to invoke the services relevant to the human feedback, e.g., filters, buffers and other tools for retrieving and managing the human feedback. Furthermore, this toolkit includes design patterns and guidelines for human-centred design.

3.4.3 Security and Privacy toolkit. The Security and Privacy toolkit provides readily available security APIs and privacy guidelines. Regarding security, the developers may define a part of their code or a standalone component that has to run on a secure enclave or that the communication between components has to use OpenSSL calls. Regarding privacy, the developers may identify datasets as containing sensitive data, thus implicitly imposing constraints on their further use. Furthermore, the privacy toolkit may also include functional tools like anonymisers.

3.4.4 Dependability toolkit. The Dependability toolkit provides software that audits the code or application components against the TEACHING dependability guidelines and/or procedures. It also provides engineering pattern implementations that the developers can invoke to ensure the dependable execution of software. For instance, in cases where the developers invoke online training approaches through the AI toolkit, the dependability toolkit may allow the code to run in multiple instances implementing a consensus model.

3.4.5 Energy Efficiency toolkit. The Energy efficiency toolkit links the code or components the user would like to run with energy efficiency services provided by the underlying layers. E.g. to run an application, the toolkit may employ energy-efficient approaches such as dynamic voltage and frequency scaling (DVFS), power mode management (PMM) or using unconventional cores such as DSP or GPUs of FPGAs. This can be done automatically or invoked by the user (e.g., "annotating" a part of the code or a component).

3.4.6 TEACHING CPSoS Applications. The TEACHING applications may comprise loosely coupled, standalone, independent components (e.g., docker images) that the TEACHING SDK builds or software that the TEACHING SDK compiles and executes.

4 CONCLUSION

This paper presented the TEACHING Project, which enables sustainable human participation in developing, optimising and overseeing intelligent autonomous systems. Through rigorous proofs of equivalence and trustworthiness, TEACHING builds CPSoS as adaptive partners in human progress.

ACKNOWLEDGMENTS

This research was supported by TEACHING, a project funded by the EU Horizon 2020 research and innovation programme under GA n. 871385.

REFERENCES

[1] Andrea Bondavalli, Sara Bouchenak, and Hermann Kopetz. 2016. *Cyber-physical systems of systems: foundations–a conceptual model and some derivations: the AMADEOS legacy.* Vol. 10099. Springer.

[2] TensorFlow Developers. 2023. *TensorFlow.* https://doi.org/10.5281/zenodo.7764425 Specific TensorFlow versions can be found in the "Versions" list on the right side of this page.
See the full list of authors on GitHub..

[3] Steve Mann. 2001. Wearable computing: Toward humanistic intelligence. *IEEE Intelligent Systems* 16, 3 (2001), 10–15.

[4] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32.* Curran Associates, Inc., 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[5] TEACHING. 2020. TEACHING Website. https://cordis.europa.eu/project/id/871385.

[6] TEACHING. 2021. D1.1: Report on TEACHING related technologies SoA and derived CPSoS requirements. https://teaching-h2020.eu/sites/default/files/docs/D1.2-TEACHING-CPSoS-architecture-and-specifications.pdf.