MINISTERUL EDUCAȚIEI ȘI CERCETĂRII AL REPUBLICII MOLDOVA
Universitatea Tehnică a Moldovei
Facultatea Calculatoare, Informatică și Microelectronică
Departamentul Ingineria Software și Automatică

ANALIZA INTEGRĂRII MODELELOR LINGVISTICE DE MARI DIMENSIUNI CU ADMINISTRAREA SISTEMELOR DE CALCUL DE ÎNALTĂ PERFORMANȚĂ

Proiect de master

Student: _____ **Dumitrașcu Marius, TIA-241M**

Coordonator: _____ **Ludmila Peca, dr. conf. univ.**

Consultant: _____ **Cojocaru Svetlana, asist. univ.**

Chișinău, 2026

REZUMAT

Dumitrașcu Marius. Analiza integrării modelelor lingvistice de mari dimensiuni cu administrarea sistemelor de calcul de înaltă performanţă. Teză de master. Universitatea Tehnică a Moldovei, Chișinău, 2026.

Structura lucrării: introducere, 5 capitole, concluzii, bibliografie (37 titluri), 3 anexe.

Cuvinte-cheie: LLM on-premise, asistent conversațional, router semantic, portal self-service, provisionare mașini virtuale, .NET, Blazor WebAssembly, Proxmox VE, Apache Guacamole, utilizatori non-tehnici, mediu academic.

Lucrarea pornește de la o problemă observabilă în multe centre academice: profesorii, cercetătorii, asistenții și studenții care nu vin din informatică au mari dificultăți să folosească resursele de calcul puse la dispoziție. Lucrul prin SSH, configurarea manuală a mediilor și sintaxa scripturilor de job descurajează aproape orice utilizator nou, iar echipamentele rămân subutilizate chiar dacă au necesitat investiții serioase. Pe partea de calcul de înaltă performanță (HPC) efectul devine și mai pronunțat, fiindcă instrumentele standard (SSH, SLURM, scripturi de job, module software) presupun un timp de învățare pe care un cercetător din chimie sau un profesor de pedagogie rareori și-l permite.

Soluția propusă în lucrare este o platformă web on-premise care funcționează ca portal self-service, cu un model de limbaj instalat în centrul de date local pe post de asistent. Utilizatorul își formulează nevoia în limbaj natural, iar platforma se ocupă de restul: alocă o mașină virtuală dintr-un pool predefinit pe cluster Proxmox VE, îi configurează accesul remote prin Apache Guacamole și îl ghidează pas cu pas la instalarea uneltelor specifice cursului sau cercetării. Platforma se așază deliberat alături de portalurile de management HPC clasice (Open OnDemand, ColdFront), nu le înlocuiește, fiindcă se adresează altei categorii de utilizatori și unui alt punct de intrare în infrastructura instituțională. Pentru cei care au nevoie totuși de cluster-ul HPC tradițional, mașina virtuală provizionată poate fi configurată cu client SLURM și mount NFS la stocarea partajată, devenind front-end pentru submisia de joburi fără linie de comandă directă pe nodurile de login.

Pe partea de inteligență artificială, soluția folosește o arhitectură dual-model cu router semantic. Tehnica nu este originală în sine, vine din literatura recentă pe cost-aware quality routing pentru LLM-uri (Arch-Router, RouteLLM, Hybrid LLM), dar e aplicată aici într-un context academic și integrată cu fluxul de provisionare a mediilor de lucru. În aproximativ 50 de milisecunde, modelul Arch-Router-1.5B (Tran et al., 2025) clasifică fiecare mesaj și alege profilul LLM potrivit: fie Qwen3 Instruct, rapid și conversațional, pentru recomandările uzuale (aproximativ 70-80% din trafic), fie DeepSeek-R1 Distill Llama 70B, pentru cererile de debugging și optimizare care cer raționament chain-of-thought (restul de 20-30%). Cele două modele principale rulează pe două noduri GPU dedicate, cu câte 96 GB VRAM și cuantizare INT8, router-ul fiind co-locat pe primul nod. Toate trei sunt servite de vLLM, cu continuous

batching și fereastră de context de 32 768 de token-uri, iar comunicarea trece printr-un gateway compatibil OpenAI operat de centrul de informatică al universității. Contribuția originală a tezei nu este tehnica de rutare în sine, ci modul în care e împachetată: rutarea, fluxul de provisionare VM, ghidul auto-generat și split-screen-ul chat + Guacamole sunt aduse împreună într-un pipeline coerent pentru utilizatorul academic fără experiență de sysadmin.

Stack-ul tehnologic ales este .NET 10, cu Blazor WebAssembly în frontend și ASP.NET Core Minimal APIs în backend, plus integrare nativă cu Microsoft Entra ID pentru autentificarea pe baza contului de mail instituțional Office 365. Pe lângă uniformitatea limbajului între cele două straturi, alegerea aduce un ecosistem matur de pattern-uri: Clean Architecture, CQRS prin MediatR, EF Core și SignalR pentru streaming real-time. În concret, partea construită cuprinde 51 de endpoint-uri REST, două hub-uri SignalR (unul pentru streamul de chat, celălalt pentru notificările de stare VM), integrarea cu Proxmox VE pe baza unui mecanism de pool management, cinci șabloane VM pentru scenariile academice uzuale și 73 de teste unitare automate.

Soluția a fost verificată pe câteva scenarii reprezentative: pregătirea unui laborator de curs cu cerere bulk de mașini virtuale și înrolarea studenților prin coduri de invitație; sesiuni individuale de cercetare cu medii GPU pentru machine learning; submisii de joburi către cluster-ul HPC pornind de la un VM configurat ca front-end. La final, teza include o analiză tehnico-economică pe trei ani care compară varianta on-premise propusă cu trei alternative concrete (cloud comercial pur, API LLM extern și arhitectură single-model) și identifică pragurile la care fiecare opțiune devine preferabilă. Pentru o instituție care reutilizează infrastructură existentă (cazul UTM), platforma propusă iese de aproximativ șaptesprezece ori mai ieftină decât echivalentul pe Azure, iar față de o variantă hibridă (VM-uri locale + API LLM extern) avantajul devine vizibil dincolo de aproximativ 1500 utilizatori activi sau în situațiile care cer suveranitate a datelor.

ABSTRACT

Dumitrașcu Marius. Analysis of the Integration of Large Language Models with the Administration of High-Performance Computing Systems. Master's Thesis. Technical University of Moldova, Chișinău, 2026.

Thesis structure: introduction, 5 chapters, conclusions, bibliography (37 titles), 3 annexes.

Keywords: on-premise LLM, conversational assistant, semantic router, self-service portal, virtual machine provisioning, .NET, Blazor WebAssembly, Proxmox VE, Apache Guacamole, non-technical users, academic environment.

This thesis starts from a problem that is easy to observe in many academic centers: professors, researchers, teaching assistants and students without an IT background struggle to use the computing resources made available to them. Working through SSH, configuring an environment by hand and writing job scripts discourages almost any new user, and the equipment ends up underused despite serious investment. On the high-performance computing (HPC) side the effect becomes even more pronounced, because the standard tools (SSH, SLURM, job scripts, software modules) require a learning curve that a chemistry researcher or a pedagogy professor can rarely afford.

The proposed solution is an on-premise web platform that works as a self-service portal, with a large language model deployed locally in the datacenter acting as the assistant. The user phrases the need in natural language and the platform takes care of the rest: it allocates a virtual machine from a predefined pool on a Proxmox VE cluster, configures remote access through Apache Guacamole, and walks the user step by step through the installation of the tools required for the course or for the research task. The platform deliberately sits alongside existing HPC management portals (Open OnDemand, ColdFront) rather than replacing them, since it targets a different category of users and a different entry point into the institutional infrastructure. For those who do need the traditional HPC cluster, the provisioned virtual machine can be configured with the SLURM client and an NFS mount to the cluster's shared storage, turning into a front-end for job submission without requiring direct command-line interaction on the login nodes.

On the AI side, the solution uses a dual-model architecture with a semantic router. The technique is not original in itself, it comes from recent literature on cost-aware quality routing for LLMs (Arch-Router, RouteLLM, Hybrid LLM), but it is applied here in an academic context and tied into the work-environment provisioning flow. In about 50 milliseconds, the Arch-Router-1.5B model (Tran et al., 2025) classifies each incoming message and picks the right LLM profile: either Qwen3 Instruct, fast and conversational, for routine recommendations (around 70-80% of traffic), or DeepSeek-R1 Distill Llama 70B, for debugging and optimization requests that need chain-of-thought reasoning (the remaining 20-

30%). The two main models run on two dedicated GPU nodes with 96 GB VRAM each and INT8 quantization, with the router co-located on the first node. All three are served by vLLM with continuous batching and a 32,768-token context window, and traffic is mediated by an OpenAI-compatible gateway operated by the university's IT center. The original contribution of the thesis is not the routing technique itself, but the way it is packaged: routing, the VM provisioning flow, the auto-generated installation guide and the chat + Guacamole split-screen are stitched together into a coherent pipeline for the non-technical academic user.

The chosen technology stack is .NET 10, with Blazor WebAssembly on the frontend and ASP.NET Core Minimal APIs on the backend, plus native integration with Microsoft Entra ID for authentication based on the institutional Office 365 mail account. Beyond keeping the language uniform across the two layers, the choice brings a mature ecosystem of patterns: Clean Architecture, CQRS via MediatR, EF Core and SignalR for real-time streaming. Concretely, what was built includes 51 REST endpoints, two SignalR hubs (one for the chat stream, the other for VM status notifications), Proxmox VE integration based on a pool management mechanism, five VM templates for the typical academic scenarios, and 73 automated unit tests.

The solution was validated on a few representative scenarios: preparing a course laboratory with a bulk request for virtual machines and enrolling students through invitation codes; individual research sessions with GPU environments for machine learning; job submissions to the HPC cluster from a VM configured as a front-end. The thesis closes with a three-year techno-economic analysis that puts the proposed on-premise variant against three concrete alternatives (pure commercial cloud, external LLM API and single-model architecture) and identifies the thresholds at which each option becomes preferable. For an institution that reuses existing infrastructure (the UTM case), the proposed platform turns out to be roughly seventeen times cheaper than the Azure equivalent, and against a hybrid variant (local VMs + external LLM API) the advantage becomes visible past about 1500 active users or in setups that require data sovereignty.

CUPRINS

LISTA ABREVIERILOR

ADR – Architectural Decision Record (Înregistrare a Deciziei Arhitecturale)

AGPL – Affero General Public License (Licență Publică Affero)

API – Application Programming Interface (Interfață de Programare a Aplicațiilor)

AWS – Amazon Web Services

BFF – Backend for Frontend (Backend pentru Frontend)

BSD – Berkeley Software Distribution (licență BSD)

CI/CD – Continuous Integration / Continuous Deployment (Integrare Continuă / Deployment Continuu)

CPU – Central Processing Unit (Unitate Centrală de Procesare)

CQRS – Command Query Responsibility Segregation (Separarea Responsabilităților Comenzi-Interogări)

CRUD – Create, Read, Update, Delete (Creare, Citire, Actualizare, Ștergere)

CUDA – Compute Unified Device Architecture (Arhitectura Unificată de Calcul pe Dispozitive NVIDIA)

DB – Database (Bază de Date)

DNS – Domain Name System (Sistem de Nume de Domeniu)

DSL – Domain-Specific Language (Limbaj Specific Domeniului)

DTO – Data Transfer Object (Obiect de Transfer al Datelor)

EF Core – Entity Framework Core (cadru de mapare obiect-relațională .NET)

GPU – Graphics Processing Unit (Unitate de Procesare Grafică)

HPC – High-Performance Computing (Calcul de Înaltă Performanță)

HTTP / HTTPS – HyperText Transfer Protocol / Secure (Protocol de Transfer HyperText / Securizat)

IaC – Infrastructure as Code (Infrastructură ca Cod)

IDE – Integrated Development Environment (Mediu Integrat de Dezvoltare)

INT8 – 8-bit integer quantization (cuantizare pe 8 biți întregi)

IP – Internet Protocol (Protocol Internet)

JSON – JavaScript Object Notation (Notație de Obiecte JavaScript)

JWT – JSON Web Token (Token Web JSON)

KV cache – Key-Value cache (memorie cache cheie-valoare a modelului LLM)

KVM – Kernel-based Virtual Machine (Mașină Virtuală Bazată pe Kernel)

LDAP – Lightweight Directory Access Protocol (Protocol Ușor de Acces la Director)

LLM – Large Language Model (Model de Limbaj de Mari Dimensiuni)

LoRA – Low-Rank Adaptation (Adaptare cu Rang Redus)

MFA – Multi-Factor Authentication (Autentificare Multi-Factor)

MIT – Massachusetts Institute of Technology (licență)

NFS – Network File System (Sistem de Fișiere în Rețea)

OIDC – OpenID Connect (Protocol de Autentificare OpenID)

OS – Operating System (Sistem de Operare)

PBS – Portable Batch System (Sistem Portabil de Programare a Joburilor)

PUE – Power Usage Effectiveness (Eficiența Utilizării Energiei)

RAG – Retrieval-Augmented Generation (Generare Augmentată prin Recuperare)

RAM – Random Access Memory (Memorie cu Acces Aleator)

RDP – Remote Desktop Protocol (Protocol Desktop la Distanță)

REST – Representational State Transfer (Transfer de Stare Reprezentațională)

SLURM – Simple Linux Utility for Resource Management (Utilitar Linux pentru Managementul Resurselor)

SPA – Single-Page Application (Aplicație de Tip Pagină Unică)

SQL – Structured Query Language (Limbaj de Interogare Structurat)

SSE – Server-Sent Events (Evenimente Trimise de Server)

SSH – Secure Shell (Shell Securizat)

SSO – Single Sign-On (Autentificare Unică)

TCO – Total Cost of Ownership (Cost Total de Proprietate)

TCP – Transmission Control Protocol (Protocol de Control al Transmisiei)

TLS – Transport Layer Security (Securitate la Nivelul Transportului)

UI – User Interface (Interfață Utilizator)

UML – Unified Modeling Language (Limbaj Unificat de Modelare)

URL – Uniform Resource Locator (Localizator Uniform de Resurse)

UTM – Universitatea Tehnică a Moldovei

vCPU – Virtual CPU (Procesor Virtual)

vLLM – engine de inferență LLM dezvoltat la UC Berkeley

VM – Virtual Machine (Mașină Virtuală)

VNC – Virtual Network Computing (Calcul Virtual în Rețea)

VPN – Virtual Private Network (Rețea Privată Virtuală)

VRAM – Video RAM (Memorie Video)

W8A8 – Weights 8-bit, Activations 8-bit (cuantizare INT8 pe ponderi și activări)

WSS – WebSocket Secure (WebSocket Securizat)

YAML – YAML Ain't Markup Language (limbaj de serializare a datelor)

INTRODUCERE

Centrele academice şi de cercetare dispun astăzi de o capacitate de calcul, care în urmă cu un deceniu era foarte greu accesibilă. Clusterele moderne combină procesoare multi-core, acceleratoare grafice şi sisteme de stocare distribuite şi permit rularea unor aplicaţii complexe din simulări numerice, analiză de date masive, învăţare automată sau modelare ştiinţifică.

De asemenea, modelele de limbaj de mari dimensiuni (Large Language Models - LLM) au atins în ultimii ani un nivel de competenţă la care pot asista efectiv utilizatorii în interacţiunea cu sisteme tehnice, schimbând modul în care expertiza este disponibilă utilizatorului final.

Capacitatea există, dar accesul efectiv la ea rămâne greu pentru o parte largă a comunităţii academice. Profesorii şi cercetătorii din domenii non-tehnice, asistenţii care îi sprijină şi studenţii care abia se familiarizează cu mediul universitar întâmpină frecvent obstacole serioase atunci când vor să-şi pregătească un mediu de lucru, cum ar fi, crearea şi configurarea de maşini virtuale, instalarea de pachete software şi aplicaţii necesare pentru un curs sau experiment, alocare şi configurare mediu de stocare, programare şi configurare de job-uri. Instrumentele tradiţionale (SSH, SLURM, scripturi shell, module software) presupun cunoştinţe care depăşesc pregătirea tipică a unui specialist din alte domenii. Rezultatul se vede în statisticile de utilizare a infrastructurii: investiţiile serioase ale instituţiei în echipamente de calcul nu se traduc în impact academic atunci când doar o parte restrânsă a comunităţii reuşeşte să le folosească.

În ce priveşte modelele LLM actuale, acestea au atins un nivel la care pot interpreta cereri în limbaj natural, pot explica concepte tehnice şi pot ghida utilizatorii pas cu pas prin sarcini complexe. Atunci când un astfel de model este instalat în centrul de date local (pentru a respecta cerinţele de confidenţialitate ale instituţiei) şi plasat într-o platformă web, devine un mediator între utilizatorul non-tehnic şi complexitatea infrastructurii. Sistemul preia descrierea în limbaj natural şi o transformă într-o secvenţă de paşi concreţi, de la alegerea unei configuraţii potrivite de maşină virtuală până la ghidarea utilizatorului prin instalarea instrumentelor cerute de scenariul de lucru, trecând prin provisionarea automată şi configurarea accesului la distanţă.

Lucrarea de faţă descrie cum a fost gândită, construită şi verificată o astfel de platformă. Soluţia propusă este un portal web, on-premise, care joacă rolul de strat conversaţional şi operaţional între utilizatorul non-tehnic şi infrastructura de virtualizare a centrului academic. Poziţionarea ei e deliberată: stă alături de portalurile de management HPC consacrate (Open OnDemand, pentru cei deja familiarizaţi cu submisia de joburi, şi ColdFront, pentru gestiunea alocărilor instituţionale), dar nu le înlocuieşte, fiindcă se adresează altei categorii de utilizatori şi unui alt punct de intrare în infrastructura instituţională. Pentru cei care au totuşi nevoie de cluster-ul HPC tradiţional, maşina virtuală provizionată poate fi configurată ca front-end, cu client SLURM şi acces la stocarea partajată, prin care utilizatorul îşi creează şi configurează joburile, fără să interacţioneze direct cu nodurile de login (maşinile pe care utilizatorii se

conectează prin SSH atunci când vor să folosească cluster-ul), pasul tradițional din care derivă, de altfel, bariera tehnică pentru utilizatorii vizați de această lucrare.

Stack-ul tehnologic, compus din .NET 10 cu Blazor WebAssembly în frontend și ASP.NET Core Minimal APIs în backend, Proxmox VE pentru virtualizare, Apache Guacamole pentru acces remote, Microsoft Entra ID pentru autentificare instituțională, a fost ales tocmai pentru integrarea naturală cu serviciile pe care universitatea le folosește deja.

Componenta cea mai distinctă a soluției este arhitectura cu două modele LLM, cu rutare semantică. Fiecare mesaj al utilizatorului trece printr-un router compact, care îl direcționează apoi către profilul de inferență potrivit complexității cererii: un model conversațional rapid pentru recomandările obișnuite și un model specializat pe raționament aprofundat pentru cererile de debugging sau optimizare. Tehnica de rutare în sine este preluată din lucrări recente din domeniu (Arch-Router, RouteLLM, Hybrid LLM) și nu este revendicată ca originală. Contribuția lucrării stă în adaptarea ei la contextul academic și în integrarea cu fluxul end-to-end de provisionare a mediilor de lucru, de la formularea cererii în limbaj natural până la mediul funcțional accesibil în browser.

Lucrarea este structurată în cinci capitole, urmate de concluzii, bibliografie și trei anexe. Primul capitol face analiza domeniului de studiu, identifică sistemele similare existente și fixează scopul, obiectivele și cerințele platformei (funcționale și nefuncționale). Capitolul al doilea trece la modelarea și proiectarea sistemului, prin diagramele UML, modelul de domeniu, deciziile arhitecturale și arhitectura modelului LLM dual cu router semantic. Realizarea efectivă este descrisă în capitolul 3, împreună cu tehnologiile folosite, integrarea cu serviciile externe (Proxmox VE, Apache Guacamole, Microsoft Entra ID) și rezultatele testării. Capitolul 4 are caracter de manual de utilizare, construit pe patru scenarii reprezentative și însoțit de o referință a interfeței. Capitolul 5 cuprinde analiza tehnico-economică pe trei ani și pune varianta on-premise propusă față în față cu alternativele de implementare în cloud comercial, prin API LLM extern și prin arhitectură single-model. La final, concluziile sintetizează obiectivele atinse, discută discrepanțele dintre specificația inițială și implementare și conturează direcțiile de evoluție.

BIBLIOGRAFIE

[1] Hudak, D. et al. (2018). Open OnDemand: A web-based client portal for HPC centers. JOSS 3(25):622. DOI: 10.21105/joss.00622.

[2] Rothwell, B. et al. (2022). Quantifying the Impact of Advanced Web Platforms on HPC Usage. PEARC '22, ACM. DOI: 10.1145/3491418.3530758.

[3] Bruno, A., Sajdak, D. (2021). ColdFront: Resource Allocation Management System. PEARC '21, ACM. DOI: 10.1145/3437359.3465585.

[4] Shu, P. et al. (2025). Survey of HPC in US Research Institutions. arXiv:2506.19019.

[5] Reuther, A. et al. (2024). Interactive and Urgent HPC: Challenges and Opportunities. WIUHPC 2024 / arXiv:2401.14550.

[6] Srivatsa, K. G. et al. (2024). A Survey of using LLMs for Generating Infrastructure as Code. ICON 2023 / arXiv:2404.00227.

[7] Zhang, L. et al. (2025). A Survey of AIOps in the Era of Large Language Models. ACM Computing Surveys. DOI: 10.1145/3746635.

[8] Vitui, A., Chen, T.-H. (2025). Empowering AIOps: Leveraging LLMs for IT Operations Management. arXiv:2501.12461.

[9] Yin, J. et al. (2024). chatHPC: Empowering HPC users with large language models. The Journal of Supercomputing. DOI: 10.1007/s11227-024-06637-1.

[10] AskHPC: A ChatBot for HPC User Support. (2025). SC '25 HUST Workshop, ACM. DOI: 10.1145/3731599.3767433.

[11] Wu, M., Zhang, Z. (2025). Maple: A Multi-agent System for Portable Deep Learning across Clusters. arXiv:2510.08842.

[12] Chen, L. et al. (2024). The Landscape and Challenges of HPC Research and LLMs. arXiv:2402.02018.

[13] Tran, C. et al. (2025). Arch-Router: Aligning LLM Routing with Human Preferences. arXiv:2506.16655. Model: huggingface.co/katanemo/Arch-Router-1.5B.

[14] Ong, I. et al. (2024). RouteLLM: Learning to Route LLMs with Preference Data. arXiv:2406.18665.

[15] Ding, D. et al. (2024). Hybrid LLM: Cost-Efficient and Quality-Aware Query Routing. ICLR 2024 / arXiv:2404.14618.

[16] Kwon, W. et al. (2023). Efficient Memory Management for LLM Serving with PagedAttention. SOSP '23, ACM. DOI: 10.1145/3600006.3613165.

[17] Qwen Team, Alibaba Group. (2025). Qwen3 Technical Report. arXiv:2505.09388.

[18] DeepSeek-AI. (2025). DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. Nature 645:633-638. DOI: 10.1038/s41586-025-09422-z.
[19] vLLM Project. (2025). INT8 W8A8 Quantization in vLLM.
[20] Apache Software Foundation. (2025). Apache Guacamole Manual: Implementation and Architecture, v1.6.0.
[21] Apache Software Foundation. (2024). Apache Guacamole User Guide.
[22] Use Cases of Apache Guacamole in Remote Work. (2024). IJCTT 72(11).
[23] SchedMD. (2025). Slurm Workload Manager - REST API, v25.05.
[24] Proxmox Server Solutions GmbH. (2025). Proxmox VE Administration Guide, v8.3.
[25] Microsoft Corporation. (2025). ASP.NET Core 10.0 Documentation.
[26] Microsoft Corporation. (2025). Blazor WebAssembly Documentation.
[27] Microsoft Corporation. (2025). Microsoft Identity Platform: OpenID Connect.
[28] Microsoft Corporation. (2025). Entity Framework Core 10 Documentation.
[29] Bogard, J. et al. (2024). MediatR - mediator implementation in .NET.
[30] PostgreSQL Global Development Group. (2024). PostgreSQL 16 Documentation.
[31] Ghent University HPC Team. (2024). HPC-UGent Documentation.
[32] GWDG. (2025). HPC Project Portal - User Perspective.
[33] Ohio Supercomputer Center. (2024). Open OnDemand Documentation.
[34] The University of Hong Kong. (2024). HPC-one Web Portal User Guide.
[35] University of Helsinki. (2024). HPC Environment User Guide.
[36] The University of Arizona. (2025). UArizona HPC Documentation.
[37] Calegari, P. et al. (2019). Web Portals for HPC: Survey and State-of-the-Art. ATOS Technical Report.