Sustainability for CoEs: Approaches, Successes & Lessons Learned

Introduction

The European Centres of Expertise for High-Performance Computing (CoEs) are research & development projects funded by the European Union. They target specific areas of science and industry that relate to, or requiring the use of high-performance computing (HPC), and are active in the development, adaptation or optimisation of HPC libraries and application codes which play an important role in their respective domains. Objectives of their software development work are to develop additional functionality required by their use cases and user communities, to adapt codes to best leverage new system architectures (such as accelerators), and to ensure that applications can scale to the next wave of HPC systems, which will provide 10^{18} floating point operations per second and consist of several hundred thousands of CPUs and accelerators (Exascale computers).

All CoEs share a fundamental goal – managing their innovation (mainly embodied in HPC application software codes) in such a way as to maximise the impact for scientific and industrial user communities. An important aspect related to that innovation management is to find ways to sustain (critical parts of) their operation after the end of the publicly funded project term.

Different CoEs naturally have their own interpretation of, and plans for, sustainability, informed by their unique circumstances and by what is considered a set of realistically achievable goals. Sustainability can, for example, relate to an aspiration to continue the whole CoE at full capacity beyond the funding period. Alternatively, it can relate to the development of training packages forming the base of commercial training offerings by CoE consortium members or others, or to monetising key innovations and methodologies of the CoE by sale or license provision to interested scientific or commercial parties. For the latter two, the establishment of a commercial arm which exists beyond the CoE end date can be an attractive proposition. The sources of funding can include public (research grants, licenses) or private sources (licensing or maintenance fees, consulting), and of course combinations of these.

This report is based on work undertaken by the HPC CoE Council (HPC3) and the European-funded project FocusCoE[1] in close collaboration with the HPC CoEs. It summarises the options for generating financial income by research-oriented consortia (such as the CoEs), discusses the approaches taken by the CoEs and their experience at a high level, presents success stories from the field for sustainable, scientific HPC code development and discusses the specific success criteria and challenges for the CoEs.

Sustainability Options for Research-oriented Consortia

Government-funded, multi-party R&D projects like the centres of excellence all face the question how to realise a persistent budget for their activities above and beyond that linked to a specific project grant. Typically, such grants are contingent on projects implementing an agreed DoA and reaching the objectives stated therein, and are firmly limited to a specified project term. Generating additional financial income to the project (or its member organisations) can be critical to sustain R&D activities

[1] FocusCoE is funded by the European Union's Horizon 2020 research and innovation programme under Grant Agreement 823964.

beyond the lifetime of the funded project, to carry out exploratory R&D in order to win follow-on grants, or to perform activities not foreseen in the DoA yet beneficial to relevant research fields, scientific/industrial communities, or to advance technology transfer.

Potential Sources of Funding

Potential sources of funding for a project or development team/community can be split into these categories:

A. Public bodies like higher education entities, research labs, government departments or operational entities (security organisations, armed forces, emergency services, …)

B. Public or commercial end-users of project results (CoE applications or libraries)

C. Resource or service providers (such as HPC centres or Cloud service providers)

D. Platform developers and integrators, and technology providers (such as Atos or AMD)

Category A sources already provide the basic funding for academic partners in the CoEs, as well as the targeted R&D funding for carrying out the respective DoA. Besides applying for funding for follow-up R&D projects, it can be possible to receive grants to aid technology transfer and support market entry for project results. A key requirement here is to avoid double-funding of specific activities, which is either ruled out in the funding conditions, or will reduce the original funding. For that reason, and since project results tend to be most suitable for transfer at the end of a project term, tech transfer and support actions usually follow after a successful conclusion to an R&D project. They do not help to increase the amount of funding available during project execution, and often, a different set of parties will carry out supporting activities compared to the original project consortium.

Category B sources employ key results from R&D projects to advance their own scientific or commercial goals, and could be willing to co-finance further application/library development or maintenance, or pay for adaptations according to their functional requirements and integration needs. The following specific options for generating revenue come to mind:

- *License fees for the use of application software or libraries:* this is most straightforward for closed-source software, where such fees can be requested from both academic and commercial users, potentially offering academic discounts. Open source software licenses can be constructed to require paid licenses for commercial use, as is done for instance for the CASTEP material sciences code (http://www.castep.org/), and for HADDOCK in the BioExcel CoE.

  Terms and conditions for paid licenses range from one-off fees for permanent use of a software product over limited-time fees which must be regularly renewed to pay-per-use or subscription schemes. They usually include the right of obtaining support, like for correcting bugs, and support requirements can include guaranteed reaction times.

  License revenues do of course depend on the number of customers and the value these customers can realise by using a software product. Academic customers pay lower prices (if they need to pay at all), and so commercial users need to be addressed to maximise license-based income. Costs incurred by the need to provide support have to be balanced against the license income, and challenges to provide such support by academic institutions (like providing the right expertise and complying to expected service levels) have to be acknowledged, too.

License revenues do of course depend on the number of customers and the value these customers can realise by using a software product. Academic customers pay lower prices (if they need to pay at all), and so commercial users need to be addressed to maximise license-based income. Costs incurred by the need to provide support have to be balanced against the license income, and challenges to provide such support by academic institutions (like providing the right expertise and complying to expected service levels) have to be acknowledged, too.

BioExcel did introduce a dual licensing scheme for the HADDOCK code which implements a novel, information-driven and flexible docking approach for the modelling of biomolecular complexes. Non-commercial users receive free access after filling in and submitting a license agreement, while commercial entities need to negotiate a paid HADDOCK license. Experience with this dual-licensing scheme has shown that the revenue generated by the commercial licenses is not sufficient to sustain the CoE's software development operation.

- *Fees for the maintenance or update of application software (SW) or libraries:* it is common practice in the SW industry to charge fees for maintenance (continuous improvements of a SW product, assistance in case of problems) or updates (between versions of a SW product offering improved functionality and/or performance), which can be separate from right-to-use licenses. The reasoning about SW licenses above applies in full here. In addition, charging maintenance or update fees is only possible in case of ongoing development of the SW applications or libraries, and when functional or performance improvements are added which are relevant to the (paying) end-user community.

  For open source software (OSS) codes, regular improvements and updates are usually provided from the developer community and available free of charge. Support in case of software bugs encountered by end-users, however, can be a valid business, since industrial/commercial customers tend to require short-term resolution which can be difficult to guarantee for OSS development projects.

- *Adaptation or integration of application software and libraries for end-users:* the complexity of implementation or use of HPC applications or libraries can open up business opportunities around consulting, integration and custom adaptation, which assist or enable customers to effectively use said HPC code. The range of services to be monetarized includes in-depth training of IT personnel, customer SW developers or end-users, integration of applications or code components with customer work environments and specific capability or performance improvements for specific customer use cases. Examples of the last two items are the selection of the most appropriate numerical models or solvers, generation of such models or meshes for specific customer problems, and ad-hoc code development to embed an application in a complex CAD/CAE workflow.

  Such "added value" services for enterprise SW frameworks from vendors like SAP, Oracle and Salesforce do constitute a large part of the enterprise computing software market. In the HPC field, providing consulting and integration services for the OpenFOAM CFD package for industrial CFD users is an established business segment, which shows that the OSS nature of most HPC codes is not a primary obstacle.

  In most cases, prospective customers will come from industrial or commercial end users, and will offer of SoWs with defined deliverables and time frames. Accumulating expertise of specific usage scenarios encountered by customers and of proven solution techniques is a key factor in establishing a sustainable revenue stream.

Specific optimisation of HPC applications or libraries for customer's systems are a potentially interesting, additional option for generating income. Reducing time to solution or the size of systems required to obtain a solution can be of significant value to end-customers, and specific consulting contracts can be concluded. One example is the POP CoE (https://pop-coe.eu/), which has a good track record of performing optimisation services for HPC applications (currently for free), with a focus on distributed-memory parallelism.

- *Case studies based on project results:* situations can arise in which commercial customers are interested in a third party investigating a scientific or business-related problem and providing results which support the customer business. One example is a case study for an innovative aircraft configuration, which could be carried out by a suitable research institute using their own expertise and SW codes and would result in performance estimates informing a go/no-go business decision. This assumes that the customer is not willing or able to license the key software and carry out the analysis itself, and avoids the need for training and support of the customer's staff.

Licensing of HPC applications and code components is also a key element for category C funding sources; resource or service providers would offer use and support for these HPC codes to their circle of customers, to be run on the resource provider's own systems. Differences to the discussion for category B include

- *Licensing for multiple/many end-users:* Resource providers usually serve multiple end-users, who will have different usage patterns and run different problem sizes. One approach can be to separately license the HPC application or code components to each end-user according to that end-user's usage pattern. Resource providers will be interested in realising "economies of scale" for popular HPC SW, by directly purchasing flexible, multi-user licenses and themselves charging their customers for license use. The "separate licensing" approach will require code providers to close agreements with each end-user (potentially) interested in said HPC software, and to themselves support these customers in using and often also installing the HPC SW. Code providers will need to find the right balance between maximising license revenue and limiting setup and support effort – judicious use of licenses issued to the resource provider might turn out the be the sweet spot, in particular if they incentivised the resource provider to themselves promote the use of the HPC SW.

  Similarly, pay-per-use licenses will likely be required by resource providers, and the code owners have to balance the need to maximise guaranteed license income (generated by fixed term licenses) with the perspective of maximising the market penetration (which pay-per-use licenses may help with, since they provide flexibility to customers and lower the entry hurdles).

  "Software as a service" is the final step towards end-user flexibility – code owners would contract with resource providers for them to enable end-users to run a specific HPC application or workflow and be billed by invocation; end-users could choose between different quality of service guarantees and prices, and the revenue would be split between the resource providers (for operating the systems) and the code owners. In an extreme model, an HPC application or library developer community might decide to operate the resources for a SaaS model themselves.

- *Update, maintenance and support:* In particular with larger resource providers, code providers could have the option of off-loading at least level—1 support and maintenance activities to be performed by the provider. Level-2 support, actual complex SW changes and updates would still be performed by the code owners, and paid through direct contracts with the resource providers.
- *Case studies based on project results:* These are unlikely to apply to this scenario.
- *Adaptation or integration of SW applications and libraries:* While end-user specific adaptations would be contracted directly with the respective users, resource providers will be interested in a seamless integration with their execution and management systems, and in particular in optimisations reducing their operating costs, such as performance improvements on specific execution systems, which would allow them to serve customers with fewer or smaller systems. This can be the subject of SW development contracts with resource providers, and due to the larger accumulated benefit to resource providers compared to end-users, can provide more significant income.

Technology providers and system integrators (category D) have in the past supported development and optimisation of important HPC application codes and libraries, of course with the objective of growing the market for their own platforms and systems. Being able to support key HPC codes with better performance or efficiency than the competition is very important, since large system procurements are decided by application benchmarks, and the utility of HPC systems to many industrial customers is often contingent on the stability and performance of a few applications.

HPC code developers can thus receive funding for porting their codes to novel processing or system architectures (like in the past for porting to GPGPUs), or for performing incremental, often low level optimisations (like for adapting to a new caching scheme). Such funding is often realised by establishing joint R&D centres funded by the commercial vendor. A good European example are the collaborations between NVIDIA and the GROMACS development team, which was followed up by a similar collaboration between Intel and the GROMACS team (https://e-science.se/2020/06/serc-intel-oneapi-coe).

As shown by the example, key codes can work with multiple technology developers or system integrators, and technical results from such collaborations are fed back into OSS codes. It should also be noted that such collaborations can effectively drive co-design of systems and architectures, since they can easily accommodate technology transfer between the academic and commercial world, and since the underlying HPC software's commercial relevancy provides an important incentive to the industrial partners to evolve their designs to match the code requirements.

Organisational Aspects

The previous subsection listed the different categories of parties which might provide funding or income, and discussed several kind of Deliverables or services to be provided by a project in return. This subsection looks at how projects or developer teams can structure the income generating efforts and the organisational aspects.

The most basic framework is that of a supplier⇔customer relationship between two organisations, with the project or code developer team being seen as the supplier of R&D services, licenses, or consulting/training, and the customer being a funding agency, public body or commercial entity. The gives and gets are specified by a formal contract spelling out the Deliverables and their characteristics, the remuneration, and IP rights. This scheme does require both parties to each be a natural person or legal entity, with the contract being written and executed according to commonly accepted legal rules. Both parties know exactly what they have to do to fulfil the contract, and it is also clear who would be liable to what extent in case of a breach of obligations.

This simple scheme does not apply to Horizon 2020 or Horizon Europe projects, such as R&I actions or CoEs. Such projects are consortia of several legal persons bound by an internal agreement to carry out a specific SoW as negotiated with a funding agency (the customer). The project partners are organised in different ways (universities, govt-funded labs, commercial entities, …), and they do have different capabilities and interests. Exploitation of project results is required, yet stays the obligation of the partners themselves, who need to collaborate even after the project to support exploitation (such as by providing access rights to results). Establishing a contractual supplier⇔customer relationship between, say, a full CoE and an industrial end-user is not possible; what can be done is to establish such a relationship between a subset of partners on the supplier side and the customer.

Closing commercial-grade multi-party contracts is complex, mainly since the rights and obligations on the supplier side have to be split amongst partners, and be adapted to the legal framework each partner operates under. Assuring commercial liability towards the customer in case of non-performance will be extremely difficult. Even if that is solved, effort and time required make closing such a contract for many of the above listed ways to increase income untenable, unless the monetary amount were very significant.

This leaves three approaches to effective, commercial exploitation:

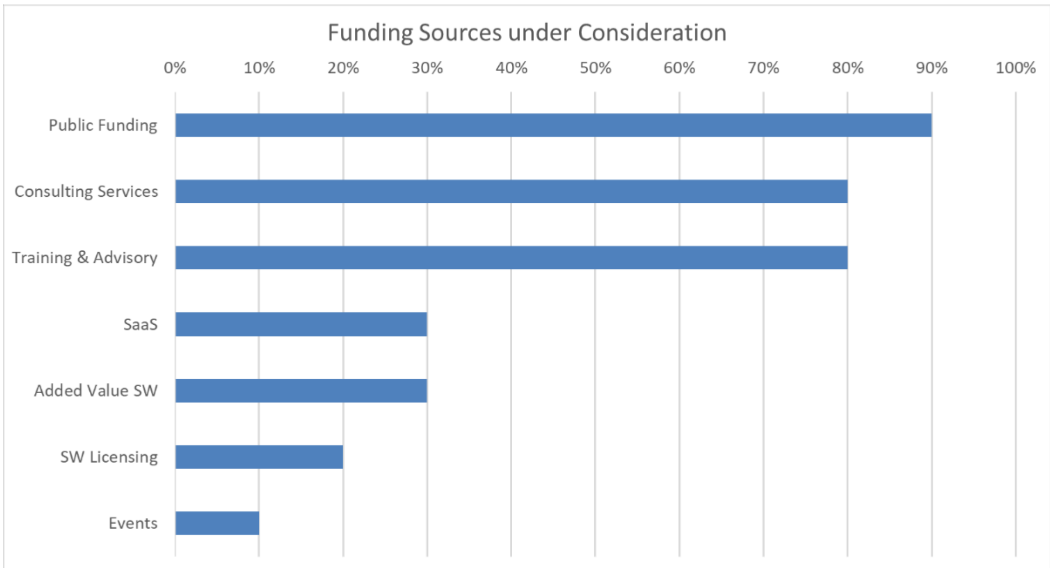
- Continue with a single partner, using the project results and potentially reimbursing project partners as per the consortium agreement
- Continue with a single partner to drive exploitation and to pursue opportunities for income, with established subcontracts to other partners carrying out R&D or providing personnel; this allows the fair pooling of competency and efforts plus haring of income, and requires a one-time negotiation of the required subcontracts
- Found a legal entity which can drive exploitation and close R&D and commercial contracts, with project partners represented and sharing income according to their contributions; this could be a commercial company, or a suitable type of association
- An alternative approach to establishing independent supplier⇔customer relations are associations with membership fees for the scientific or commercial customers. These would receive access to otherwise confidential information or codes, services like consulting or software maintenance, or influence in decisions that can impact their business. Classic example of this are certain standards bodies, where access to technical discussion bodies or voting rights and sometimes even access to relevant documentation is dependent on paid membership. One example from the HPC world is the OpenMP ARB (architecture review board). In general computing, the Linux foundation implements a similar scheme. The proceeds from membership fees support the operation of the parent organisation, and fund services for the paying members. It should be noted, however, that in case of the two examples, the development of software (OpenMP compilers, the Linux OS codebase) takes place outside the organisation (standards body), it is performed by the individual members.
- Finally, the size of prospective customers does play a role: larger companies often have sizeable R&D departments and budgets, and are able to engage with research partners on innovative and therefore somewhat risky projects. They also usually have the IT capabilities to

effectively run complex HPC codes and workflows, and can train key personnel to become proficient with such codes. Small and medium enterprises (SMEs) mostly lack these capabilities: they can engage to solve specific business problems, yet can and will not take risks, and their IT infrastructure does not include HPC systems or expertise. This makes SMEs a difficult target for CoEs – the European Union has funded the Fortissimo line of projects (https://www.fortissimo-project.eu/ and https://www.ff4eurohpc.eu/) which engaged in small, business-oriented HPC projects with SMEs, and the recurring experience is that such companies value "one stop shop" consulting offers which directly assist the end-user in the performance of complete (HPC-based) analyses and deliver results that directly feed into the customers product development.

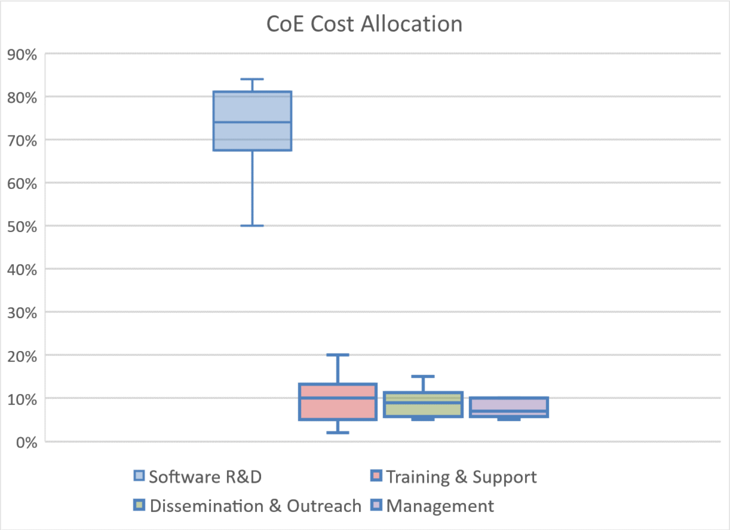
Approaches Chosen by CoEs

The FocusCoE project did support the sustainability planning of all CoEs via a variety of actions, including training events with experts in the field of innovation management and business development and interactive workshops. The HPC CoE Council also organised a working group which collaborated with the CoEs; one important outcome is a survey of the plans and experiences regarding innovation management, business development and sustainability. This survey was conducted in the second half of 2021.

The figure below shows the funding sources under consideration by the CoEs – multiple answers were possible, and so the numbers do not add up to 100%. It is noteworthy that 90% of the projects were expecting follow-on public funding, with 20% assuming that their operation would be fully funded from public sources. Amongst the commercial sources of income, consulting services and training lead by far; software licensing is only considered by 20% of the projects, with half of them also thinking about dual licensing schemes. Provision of software as a service (SaaS) and specific, licensed add-ons to software are considered by 30% of the CoEs.

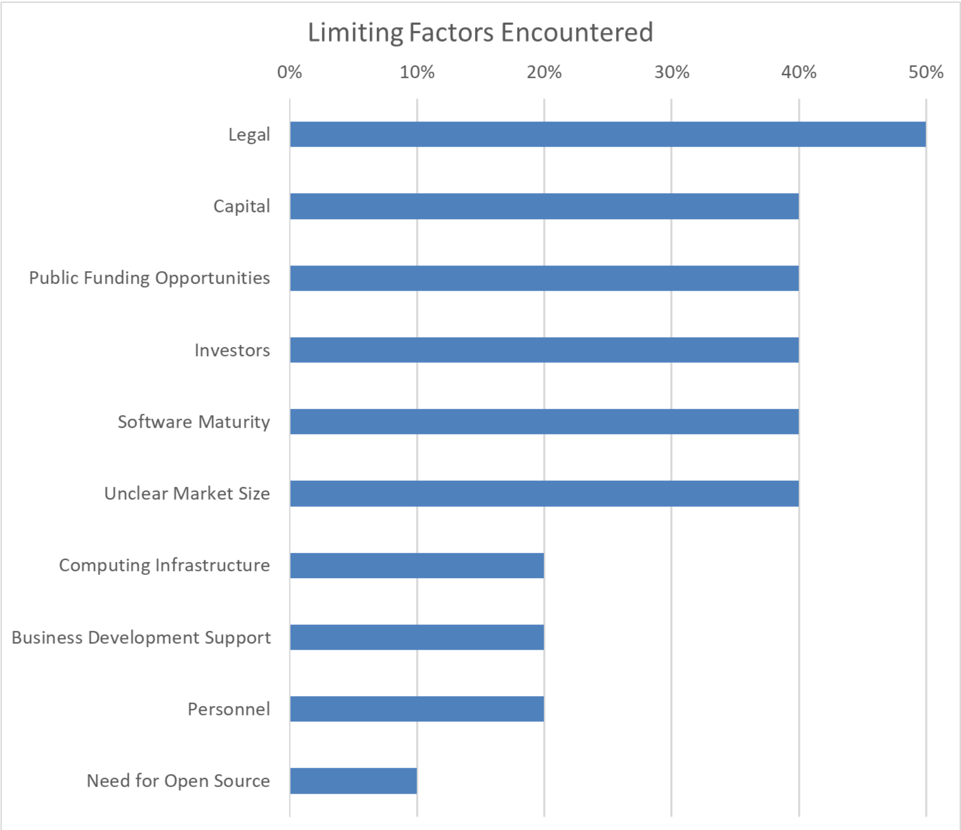


The cost allocation for the CoEs across activities is illustrated by the next figure, with the second and third quartiles indicated by the box boundaries, the average shown as a horizontal line, and outliers depicted as "whiskers" (vertical bars).

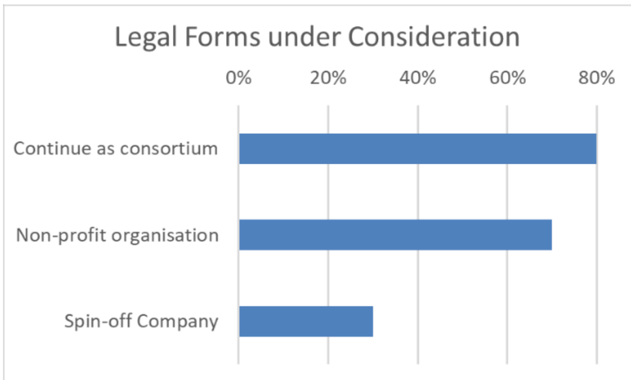


Research for and development of the CoE software clearly accounts for the lion's share of costs (73% on average). Training and user support come in second, with an average of 10%, followed by dissemination and outreach at an average of 8%. This might indicate a problem, since commercialisation of IP, services or software will likely require a higher amount of dissemination (indeed the need for marketing of the value propositions).

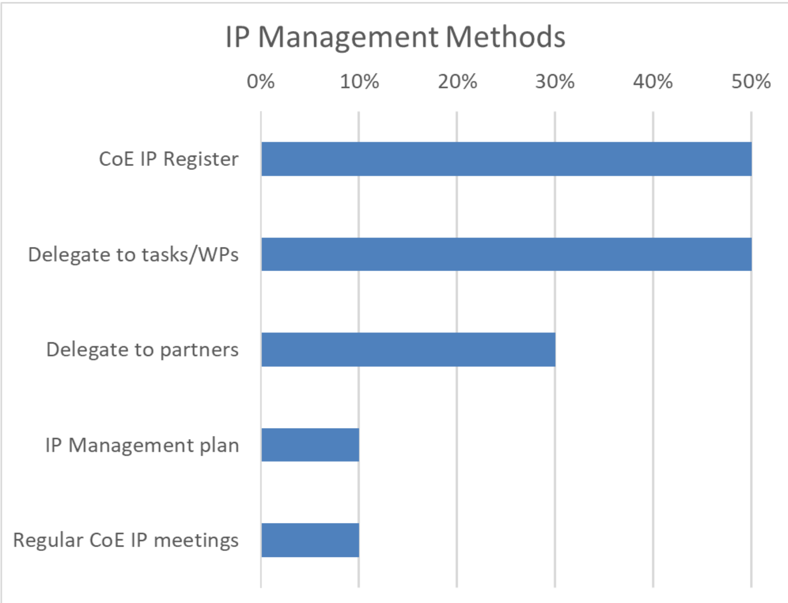
Asked about which factors have been limiting their business development and sustainability activities, legal and funding constraints, lack of investors and insufficient clarity about potential market sizes are mentioned most often (see figure below). Lack of qualified personnel was only mentioned by 20% of the respondents, and only 10% see the need to produce open source software as a limiting factor.



70% of the CoEs consider founding a non-profit organisation, and 30% are thinking about a commercial spin-off. In addition, 80% expect to continue as a consortium after the end of their term, probably anticipating follow-on funding.



Exploitation of project results to generate funding presumes alert and reliable IP management practices, which identify and boost valuable innovation in the project. The figure below summarizes the CoE practices in this respect. Half of the CoEs keep a central IP register (see figure below), yet only 10% have an IP management plan or regular IP meetings. 50% of the CoEs delegate IP management to their work packages or tasks, and 30% delegate it to their partners. The lack of central IP planning and management might indicate a problem, since there is the danger to miss recognising value propositions which combine outputs and results from different work packages or partners.



Success Stories

Open source software (OSS) has been spectacularly successful in several fields, and scientific projects have matured into large, sustained organisations which successfully develop and maintain software with a huge numbers of users and customers. A prime example is of course the Linux operating system. The paper on “How to support open-source software and stay sane” (Nowogrodski 2019) discusses factors for the success of generic OSS.

This report is specific to HPC software and to the CoEs, and thus examples from the HPC field are more relevant than those for general-purpose computing. Below, we present four such success stories, all driven by European partners.

BioExcel Enterprises

BioExcel is the European CoE for Computational Biomolecular Research. The centre significantly contributes to the development of key applications which are widely used in science and industry (like GROMACS and HADDOCK), and provides extensive support to these communities. The centre evaluated several approaches towards achieving long-term sustainability of its operations. It was recognized that core, fundamental scientific research on Exascale capabilities of codes will need substantial and continuous support from public funding grants. Still, there are opportunities for commercial activities which can supplement the research work. In order to facilitate such activities, the centre registered a legal body, BioExcel Enterprises (https://bioexcel.enterprises/), as a vehicle to engage with third parties looking for support with the development, deployment and usage of BioExcel CoE tools. BioExcel Enterprises was registered as an economic association in Sweden, which is a form of a cooperative. This is an open and inclusive legal form which allows diverse stakeholders to participate in its operations. BioExcel Enterprises provides a framework for promotion of members' products and services, establishing contacts with interested customers, setup and management of projects. The association's core coordination activities are managed by a small operative team. Projects are run in a standalone manner (in terms of personnel and budget). A small overhead from each project is taxed to support administration of the association.

GROMACS

GROMACS (https://www.gromacs.org/) is one of the most popular applications for Molecular Dynamics simulations in science and industry. In addition to the core compute engine, the application includes a rich collection of tools for pre- and post-processing and analysis of results, and ports to many different execution platforms are available, including native support of GPGPU accelerators. GROMACS is primarily designed for biochemical molecules like proteins, lipids and nucleic acids that have a lot of complicated bonded interactions, but since it is extremely fast at calculating nonbonded interactions (that usually dominate simulations) many groups also use it for research on non-biological systems, such as polymers.

GROMACS development has started over 30 years ago at the University of Groningen in the Netherlands. Current development is led by the Science for Life Laboratory in Stockholm, Sweden; while the core developers are mainly in the Stockholm region, GROMACS has many contributors worldwide. The code is open source and freely available under the GNU Lesser General Public License (LGPL), with the copyright being held by the GROMACS authors. Source code development is currently hosted on GitLab (https://gitlab.com/gromacs/gromacs/). The development process is very transparent. User feedback is continuously collected, and the most requested features are implemented to the extent that available effort allows.

The openness of the development process allowed the creation of a strong community over the years. The very large user base (est. over 10,000) has helped with raising awareness about the software. GROMACS is part of many benchmarking suites used in the procurement processes by HPC centers. The popularity has attracted research funds for its development. Importantly, hardware vendors such as Intel, NVIDIA and others are supporting the project. The vendors want to make sure that the code can run efficiently on their platforms and are willing to contribute engineering efforts and financial support.

GROMACS is a good example of achieving sustainability of software projects. Key milestones along the sustainability road have created a virtuous cycle: open source→open development process→collecting/implementing user feedback→growing the community→attracting the attention of funding agencies and businesses.

OpenFOAM

OpenFOAM (https://openfoam.org/) is the leading free, open source software for computational fluid dynamics (CFD). It was first released in 2004 under the GNU General Public License (GPL), yet its roots go back to code developments at Imperial College London, UK in the 1990s. At that time, the market for CFD applications was dominated by commercial codes developed and marketed by independent software vendors (ISVs). OpenFOAM ownership and governance have gone through several hands, including the system provider Silicon Graphics International and the French ISV ESI Group. Since 2011, the IP rights to the OpenFOAM code reside with The OpenFOAM Foundation, which since 2014 operates from the UK. The foundation has concluded contributor's agreements with the main code developers (including the original author), enabling it to safeguard and enforce the GPL v3 license conditions. Funding of the foundation comes from a wide variety of industrial entities (primarily end-users) and academic organisations (https://openfoam.org/supporters/), which buy annual maintenance plans in the € 5,000 – € 20,000 range. Total costs for maintaining the OpenFOAM code base in 2022 is projected to be € 250,000.

OpenFOAM has quickly attracted a large set of end-users from academia and industry, including leaders in the automotive industry (https://www.esi-group.com/customer-successes/audi-maximizes-the-driving-range-and-acoustic-quality-of-the-new-e-tron-using-esi-virtual-prototyping-solutions). As shown by the European Union funded series of Fortissimo projects (https://www.fortissimo-project.eu/ and https://www.ff4eurohpc.eu/), OpenFOAM is a highly popular code for CFD simulations by small and medium enterprises. Reasons for the success are the large functional scope of the code, its continuous improvement (for instance by inclusion of new, more efficient solvers), and the close integration with open source post-processing and visualisation tools. In particular for smaller companies, the free availability for use is a key factor, since it lowers the hurdles of starting with the use of CFD.

An important factor in the industrial uptake is that several ISVs have created their own versions of OpenFOAM, for which they provide paid, industrial grade support, or commercially licensed add-on products. These include ESI Group (https://www.openfoam.com/), Engys (https://engys.com/), and ICON (https://www.iconcfd.com/) – the CFD solver parts are in all cases freely available under the OpenFOAM GPL license. In addition, many engineering consulting companies are providing consulting or support to Open FOAM end users.

In summary, OpenFOAM's success is to a large part the result of the same factors as for GROMACS: open source→open development process→collecting/implementing user feedback→growing the community→attracting the attention of funding agencies and businesses. One important additional factor is the availability of paid, commercial grade support and consulting for OpenFOAM users via a variety of channels and entities.

VI-HPS

The Virtual Institute — High Productivity Supercomputing (VI-HPS, https://www.vi-hps.org/) was founded in 2007 by European and US universities and research organisations that were leading the fields of HPC performance analysis tools and methodologies. Its mission is to develop and improve integrated state-of-the-art HPC programming tools which will assist domain scientists in diagnosing coding errors and optimising performance and energy efficiency of their applications. It has 14 members today, and was funded initially by the Helmholtz Association of German Research Centres; after that, the virtual institute is funded by contributions from its partners. The majority of POP CoE partners is participating in VI-HPS.

The institute is organising a wide variety of regular events targeting the performance tools community and more importantly HPC application developers; these include the 3-5 day VI-HPS Tuning Workshops, which occur at least twice per year and instruct participants in VI-HPS tools and offer hands-on experience and expert coaching assistance on using the tools on real-world codes. In addition, there are the yearly workshops on programming and performance visualization tools which discuss new approaches to assist developers in analysing, understanding, and optimizing programs for extreme-scale platforms.

The performance analysis tools field is fragmented, with a significant number of academic partners and a few commercial entities developing their own line of tools, traditionally without paying much regard to interoperability or composability. VI-HPS is playing a key role in bundling the academic tools and methodology development in Europe and the US, and in driving towards more universal interoperability (such as realised in the Score-P measurement framework). It also acts as a "one stop shop" for application developers, who have access to first-class training materials and the full scope of tools. Finally, the hands-on tuning workshops (now in their 41st incarnation) have jump-started the use of the VI-HPS tools, and more importantly of sound performance engineering practices, amongst many hundreds of developers.

In summary, VI-HPS is a good example for a sustained & successful collaboration in a key technical field, with significant impact on making the results of mainly publicly-funded tools development more sustainable and increasing the circle of end-users of this technology.

Lessons Learned on Commercialisation of CoE Application Codes and Libraries

In the course of the discussions with the CoEs, during workshops organised by FocusCoE (D4.2, D3.3), through an analysis of the CoEs innovations (D5.5, innovation radar), and from experience in meeting potential customers during sectorial events (D3.4), a number of recurring factors emerged, that either favour or, more often, challenge a sustainable commercialisation of research results through direct interaction with science and industry.

Firstly, when considering a licensing or consulting business based on HPC software, the size of the customer base plays a decisive role. This favours large, mature codes with a relatively broad area of application, such as computational fluid dynamics (CFD) codes (see for instance the previous subsection on OpenFOAM). Commercial customers will have more trust in established codes with a proven track record of delivering reliable results and the provision of support before they invest in it - financially by paying license or consulting fees, but also in expending effort to adapt their workflows and processes and train their workforce. In addition, they value ease-of-use and mature, product-quality software; these two aspects do not normally have the highest priority in research projects, and the hardening and validation of codes required during productization requires a very significant effort.

While open source codes have a definitive up-front advantage since they do not require investments in license fees, they may fall short of commercial software with respect to the other qualities unless a proven, reliable and sustainable structure for maintenance and support is set up (see the previous subsection on GROMACS). In the Fortissimo projects it became clear that open source software (notably OpenFOAM) can be successfully applied by SMEs; however, there was no overall tendency to prefer open source software, and the required adaptation and consulting was financed by the projects, not by the industrial end-users.

Moreover, the applications and libraries offered by many (but not all) CoEs are often more specialised than a “generic” CFD solver, and they also address highly advanced, leading-edge HPC use cases; as a result, the considerations above imply restrictions of the addressable market. No reasonably complete market studies are available, yet again, the experience from the Fortissimo projects gives an indication: Even in a situation where all expenses of taking up HPC applications and workflows for SMEs are covered, the number of submitted proposals typically ranged between 70-100 per call. This implies that reaching out to commercial customers that could potentially benefit from the CoE codes and base continued operation of CoEs on the revenues achieved is not a “home run” – at least with SMEs.

A factor specific to the current CoEs is the funding agency’s emphasis on adapting applications and libraries to run efficiently on future Exascale systems. Achieving this objective will create significant value in the future, but the value of Exascale-ready codes for industrial end-users is limited at this juncture. Current practice across industrial and commercial use cases is to run on much smaller systems such as Terascale (10^{12} Flop/s) or Petascale (10^{15} Flop/s), and to fill larger systems by running multiple independent instances.

Given the relatively short-lived nature of project like the CoEs (approx. 3 years compared to the decades-long live span of successful HPC software – cf. the OpenFOAM and GROMACS examples), it is difficult to build the required visibility and trust for software with a small user base, and thus guarantee stable long-term development. Put differently, CoEs have to rely on long-standing governance structures for code development and maintenance independent of short-term project terms to become a successful market player.

Acquisition of customers can take a long time and requires specific (non-scientific) expertise (on top of technical/scientific skills), in particular for the highly complex simulation codes typically developed by the CoEs. This further cuts opportunities, and industrial customers are more likely to establish a business relationship with a (permanent) commercial organisation – or an organisation with strong commercial links[2] –which “speaks their language” rather than a transient project which ends soon with an unclear long-term perspective.

Assuming that there is initial interest from the commercial customers in technological offerings of a CoE, additional factors have to be considered before a business-to-business relationship can prosper:

- Time-scales must be aligned, with customers requiring short-term, assured availability of key staff, which in scientific institutions often depends on immutable project or paper submission schedules.
- Performing services for a commercial customer may be at odds with the personal goals of scientists, since doing so does not contribute to a scientific career, in particular as results are often confidential and cannot be published. It has also proven to be difficult for scientists to temporarily transition to industry, or even to pursue a “parallel career” in both worlds.
- Academic employment contracts are usually temporary with immutable terms. This often leads to the situation of the “unit of currency” for collaborations with industry being the duration of an entire PhD work. Such time scales might well be at odds with the expectations of industrial customers. Institutions should keep close contact with former employees

[2] This has been the case for the two OSS success stories on GROMACS and OpenFOAM above.

transitioned into businesses (perhaps start-ups), because they can come in handy for work with industry.

- Successful business development and customer acquisition requires specific "talents" and expertise which are not in the key skills set for domain scientists or expert software developers. It is important for research projects to seek expert advice and involve persons with the requisite expertise during their lifetime, and institutions active in such projects should seek to employ and retain such persons long-term.

- The business value of the CoE software or service to the customer must clearly exceed the incremental costs incurred by the CoE; if that is not the case, the CoE will not be able to charge a price sufficient to avoid losing money.

On top of these, there are specific legal and organisational factors in play for the CoEs: they are no legal entities that can do business, and founding a legal entity is a necessary first step if the business case is not limited to a single project partner. In addition, some scientific institutions exclude any commercial activities without clear scientific value.

Much of what has been said above does apply to potential scientific or academic customers for CoE codes or services; naturally, it will be much easier for CoEs to address scientific customers, and industrial aspects like ease-of-use, integration with engineering workflows and such would not apply. But that is offset by the almost universal lack of sufficient funding to pay for software licenses or maintenance/development amongst academic and scientific partners.

References

(Nowogrodski 2019) Anna Nowogrodzki; How to support open-source software and stay sane, Nature, July 1st, 2019; available at https://www.nature.com/articles/d41586-019-02046-0

(HPC3) The European HPC CoE Council, https://www.hpccoe.eu/hpc-coe-council/

(FocusCoE) Concerted action for the European HPC CoEs, https://www.hpccoe.eu/about/.

(D3.3) FocusCoE Deliverable D3.3 "Report of the actions towards SMEs", September 2021, https://www.hpccoe.eu/download/

(D3.4) FocusCoE Deliverable D3.4 "Report on the sectorial communication activities", March 2022, https://www.hpccoe.eu/download/

(D4.2) FocusCoE Deliverable D4.2 "Final Training Report", March 2022, https://www.hpccoe.eu/download/

(D5.5) FocusCoE Deliverable D5.5 "Final Innovation Management Report", March 2022, https://www.hpccoe.eu/download/

(Innovation radar) Overview over the HPC CoEs innovations listed by the EU Innovation Radar, https://www.hpccoe.eu/2021/03/22/list-of-innovations-by-coes-spotted-by-innovation-radar/

Acknowledgements

Rossen Apostolov, KTH

Anne-Bernard Bedouet, SICOS BW

Guntram Berti, scapos AG

Joachim Biercamp, DKRZ

Joan Farnós, BSC

Jon Gibson, NAG

Elena González, ATOS

Hans-Christian Hoppe, scapos AG

Francisco Javier Nieto, ATOS

Nicholas Laver, UCL

Guy Lonsdale, scapos AG

Emily Lumley, UCL

Teona Macharadze, University of Stuttgart

Hugh Martin, UCL

Luisa Neri, CNR NANO

Xavier Salazar, BSC

Julien Thélot, CEA

Mariano Vazquez, BSC