Optimising development process and software maturity through eScience partnerships

Dieter Kranzlmüller
Leibniz Supercomputing Centre (LRZ)
Garching n. Munich, Germany
Email: kranzlmueller@lrz.de

Matti Heikkurinen
MNM-Team
Ludwig-Maximilians-Universität München (LMU)
Munich, Germany
Email: heikku@nm.ifi.lmu.de

Abstract — **Computational modelling is a crucial tool in numerous basic and applied research domains. Running the simulations at unprecedented scales on supercomputer systems represents an important catalyst for improving the performance and maturity of the software. In this paper, we present a conceptual model justifying the use of software scalability as a viable proxy indicator for the maturity of the software and its development process. We also present two approaches – workshop and partnership – that allow supercomputing centres to play a more active role as partners instead of providers during the development and improvement of modelling tools. This is confirmed by the scalability results achieved.**

Keywords: software development; computational modelling; supercomputing; research; software engineering; IT service management

I. Introduction

The process that takes algorithmic results from various basic research activities and gradually turns them into software-based, infrastructure-like services running on high-end possibly interconnected, computational systems is an important part of computational science. The requirements on the working practices in the opposite ends of this process are quite orthogonal, which may lead to poor alignment of goals and incentives between researchers and computational service providers. In the (basic) research domain, repeating the previous results achieved before in the exact same form (similar data, same tools, same environment and so on) is rarely of interest. In contrast, with *"infrastructure-like"* services uniformity, repeatability and high volume of use are indicators of value. Paradoxically, when scaling these models to high-end supercomputing environments, the work addresses both of these areas: the accuracy of the models can be verified much more reliably, while at the same time the scaling challenges expose weaknesses in the implementation, fixing of which make the software package more suitable for routine production use.

The basic research activities need to minimise the "friction" between the advances made in the theoretical explanations of the phenomena and the computational models that are used to verify theories. As a result, the first versions of the modelling software tend to be implemented by the theorists themselves, using either *ad hoc* development processes or perhaps a variation of personal agile development geared toward rapid prototyping. This imposes clear limitations on the degree of formalism that can be applied to the software development and documentation. Even if the developer of the computational models has a strong software engineering background, many of the formal methods used to improve the maturity of the software development process would have a fairly low (possibly negative) return on investment. The challenges encountered – and met – during the scalability challenges may represent one of the clearest indications that motivate developers to adapt more mature software engineering practices. For this reason, we assume that the improvements in the scalability of the software suite represent a usable proxy indicator for improvements in the development process. While obviously not 100% accurate representation of the maturity of the software development process, the advantage of this indicator is that it is possible to measure in a uniform way, independent of the specific development approach used in an individual software package.

II. Characteristics of the Ecosystem

A researcher working full time implementing software corresponding to the latest theoretical model might not see any short-term quality or productivity improvements through adoption of formalized, mature software development methods. It is likely that during active development phases the key parts of the software are so closely entwined with the theories being studied that (in terms of the developer him/herself) the software is in practice self-documenting. Furthermore, if the software is most likely to be discarded (due to fundamental changes in the theory being actively tested, for example) before there is a need to involve more than a handful of developers, extensive formal documentation can seem like a waste of time.

However, some of the results of such projects are retained for longer periods of time. They may also end up being relevant in more settings than initially assumed, and some of them will thus (eventually) get reused by other researchers. Often this reuse starts as a fairly informal sharing (including copy-pasting) of code between collaborating researchers, but demand for packaging the software into formal distributions is likely to emerge gradually. This will typically lead to self-organisation of the user/developer communities that formalise the documentation, development and distribution practices, using tools and services such as GitHub [1] or Elsevier's SoftwareX [2].

Sustaining this community-driven maturing process is often not straightforward: transitioning from the "pure" research funding to what is essentially – at least partly – product development requires adopting new business models. The ways funding of these development efforts is secured is very different compared to basic research. Instead of measuring the progress through journal impact factors and number of peer-reviewed papers, the metrics need to be linked to number of users, quality of support (with e.g. speed of closing support tickets as a proxy indicator) and – usually anecdotal – evidence of the scientific contributions enabled by the software product or service. It is exceedingly rare that the funding agency that funded the initial work and has the best understanding of the expertise of the original innovator would be able to fund both types of activities, hence a structure or organisation that acts as a mediator or fulcrum in this interplay between research and development is needed.

III. SUPERCOMPUTING CENTRES AS FULCRUMS

As discussed above, the research and e-Infrastructure (or Cyberinfrastructure, as it is known in the US) funding philosophies tend to approach the "theory to computational service"-pipeline from the opposite ends. In considering the priorities of operational e-Infrastructure, the key is identifying the most often used components and measuring success – for the most part – based on the volume of use. Historically the interest (and de facto sustainable funding) has extended gradually from supporting the basic, Internet-like connectivity to more and more complex computing and data services. As an example, this development is evident when reviewing the table of contents of the White Papers of the e-Infrastructure Reflection Group [3] that have focused on areas where developments in technologies and their use cases require policy-level action. The development starting form 2003 illustrate the history of the current "layered" reference model used in the European e-Infrastructure policy work [4].

The challenge with this development path is that the maturing process starts from the opposite ends of the e-Infrastructure service and technology stack. The maturing applications should eventually "meet" the new e-Infrastructure services, but this is challenging as the developments on these two areas often proceed independently (outside specific "spearhead" applications that form the basis of the use cases supported by the new e-Infrastructure services). Without in-depth knowledge of the planned improvements of the e-Infrastructure, applications may end up with inherent limitations that prevent them from utilising all the possible benefits of the new top-level services. Similarly, the development efforts behind the standardised interfaces to provide access to supercomputing and other advanced services may not be sufficiently informed about the requirements of the existing application software solutions and their user communities. The decoupling of the typical funding sources further complicates this, as the communities lack organic meeting points (conferences, cross-project events organised by funding agencies etc.).

However, there is one area where basic research and advanced service provision meet every day: general purpose supercomputing centres. Typically such a centre supports tens or hundreds of applications and use cases, and successful day to day operations need to be based on a broad consensus on what interfaces and solutions should be supported. As an example, Leibniz Supercomputing Centre's application mix includes:

- Computational Fluid Dynamics: Optimisation of turbines and wings, noise reduction, air conditioning in trains
- Fusion: Plasma in a future fusion reactor (ITER)
- Astrophysics: Origin and evolution of stars and galaxies
- Solid State Physics: Superconductivity, surface properties
- Geophysics: Earth quake scenarios
- Material Science: Semiconductors
- Chemistry: Catalytic reactions
- Medicine and Medical Engineering: Blood flow, aneurysms, air conditioning of operating theatres
- Biophysics: Properties of viruses, genome analysis
- Climate research: Currents in oceans.

Sharing an operational production system between all these stakeholders requires balancing not only the needs of these different research activities, but also the commitments required by the routine, sustained services. Thus supercomputing centre is at the same time both an advanced service provider and a testing ground for the latest (and thus inherently somewhat immature) computational models and other software innovations. This continuous negotiation/consultation process allows supercomputing centres to serve as fulcrums and knowledge exchange channels for broader communities of application developers. As a result, e-Infrastructure service providers reap the benefits in terms of speed and efficiency of the maturing process of the application software – a benefit that is not limited to the supercomputing centres themselves.

In more detail, the key to the fulcrum role lies in the fact that the "grand challenge" supercomputing applications necessitate adopting innovative, experimental approaches both in the modelling software itself as well as the hard- and software infrastructure used to run it. Only by looking at all of these parts together, it is possible to achieve results that exceed both quantitatively and qualitatively the current state of the art. However, typical supercomputing centre is at the same time entrusted with provision of services that have very rigorous, long-term quality of service requirements (e.g. long-term archival of digital cultural heritage artefacts) that cannot be disrupted. The ability to let these two modes of operation – experimental and e-Infrastructure – coexist mean that top-level scientific computing services automatically play a crucial role in the maturing process of scientific software.

In this paper, we present the traditional model of supercomputing application development and two complementary models that leverage this “dual role” of more efficiently to facilitate and speed up the transition of theoretical models into mature computational services. The first of these advanced models is based on workshops arranged on location at the supercomputing centre. The PiCS [5] model represents a longer-term partnership where the computing centre provides a forum where computational modellers and software engineers can develop a common vision for optimal approach to software maturity and requirements for the underlying e-Infrastrucutre. We also present case studies that illustrate these models in action.

IV. Traditional Approach

The traditional approach to scaling up software to top-level computing systems is closely mimicking the general peer-review process. A software developer (i.e. researcher) attempts to prove that his project represents the best return on investment for resources on systems such as supercomputers that are either unique or have demand that outstrips the supply. One of the key arguments in this “formal proof” – in addition to the potential contribution to the research question itself – is the maturity of the software in question, especially in the sense that it can utilise the resources of the top-tier systems efficiently (scalability).

These claims by the researchers are inspected by panels of experts, who are encouraged to discard in their evaluations any knowledge of skills, motivations, previous experience, related initiatives (at the centre or among centre’s clients) that are not brought up in the application itself. After the proposals have been evaluated, the top-ranked ones are given access to the system. The access is based on an account with a quota (that specifies the maximum amount resources to be used etc.) that also gives access to support functions (ticketing system and contact information of the helpdesk) that assist in solving problems related to running software efficiently in the new environment. In case of mature software solutions, this approach is probably necessary to provide equal and fair access to the resources based prioritisation on scientific merit. However, in situations where the research question would require using modelling software that is not yet proven to be on the highest maturity level, the inherent delays of the traditional approach can be problematic. For example, the delay can mean that the team developing the software gets temporarily reassigned to other tasks, making them less effective in tackling the issues as they emerge in the large-scale systems when they eventually receive access.

In the case of PRACE allocations [6], the review process takes place every 6 months for major projects and 4 times a year for smaller scale preparatory access. This means that even if the first application for resources is successful, there will be a considerable (up to a year) delay between the moment when the need for the large-scale systems is identified and the software can be tested in the top-level systems. The cycle for the preparatory access [7] is shorter (evaluations taking place every three months) and thus halves the delay before the basis adaptation of the software to the supercomputing environment can be started. However, some of the issues with the software become apparent only once the problem size and allocated resources are scaled up sufficiently.

Less obvious, but potentially more insidious problem with the traditional approach is the inherent positioning of the parties: the supercomputing service providers are positioned as “gatekeepers”, with the task of keeping all but the “safe”, independently vetted top-applications out of the system. Conversely, the hopeful users-to-be have at least a theoretical incentive to downplay any known issues with the application code. This application-period situation may make some of the researcher less inclined to reach out and access the expertise of the top-level e-Infrastructure experts. In the end, the “time to results” may end up being longer than necessary.

V. First Step: Joint Workshops

To address these shortcomings, Leibniz Supercomputing Centre of the Bavarian Academy of Science and Humanities (LRZ) initiated “Extreme Scaling Worshop” [8] in July 2013. Projects were invited to work together with the supercomputing experts to port their application codes to SuperMUC [9] in a way that utilises efficiently the whole system. The workshop was targeting applications that were already tested in relatively high-end supercomputing environments, thus the application software solutions were relatively mature. However, the goal of the workshop was to scale up to a system 4.5 times the size previously attempted anywhere with the applications in question.

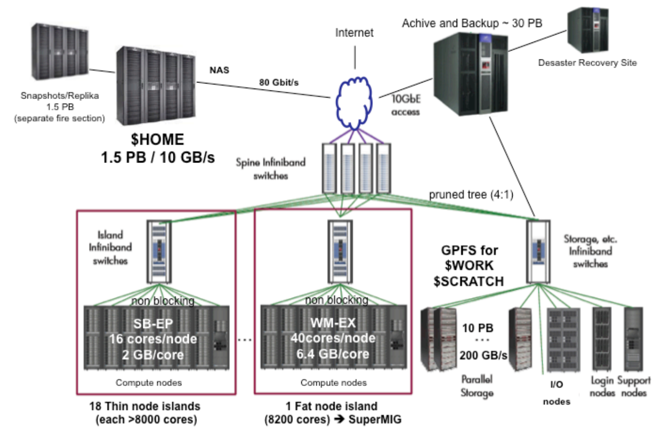

Figure 1. SuperMUC System Architecture

Despite the high level of familiarity both parties had with the environments, applications and tools, the results of the event still highlighted the advantages of the concentrated joint activities: about half of the applications reached the ambitious goal of using the whole SuperMUC system (depicted in the Figure 1 above) efficiently, while almost all of the remaining ones reached the halfway milestone (65 536 cores). In addition, the workshop allowed the application developers and supercomputing centre staff to test very rapidly different optimisation approaches that both made these excellent scaling results possible and resulted in some cases (Gromacs software) in additional 10-15% performance gains through fine-tuning some operational parameters.

 ISBN: 978-1-61208-432-9

The basic outline of the workshop was based on the following model [10]:

- The entire SuperMUC was reserved for the workshop duration of 2.5 days, with 0.5 days reserved for initial testing and 2 days for the execution of the scalability challenges
- LRZ provided automatic tools to automate compilation, submission and validation of application software and its results. This played a key role in making testing different applications in quick succession possible
- Intensive “boot camp” approach was successful in creating in-dept knowledge that the participants could pass on further. As an example, follow-up activities led to an application “performance world record”: Seissol [11] software was executed at close to 1 Petaflop/s (i.e. almost 1/3 of the theoretical SuperMUC maximum).

The summary of the increases in the number of cores the software suites can effectively use are presented in the table 1 below. Due to the successful execution of the workshop, it has become a permanent event organised by LRZ on annual basis.

Package	Description	#cores reached	TFLOP/island	TFLOPS (max)
Linpak	Top500 benchmark	128 000	161	2560
Vertex	Plasma Physics	128 000	15	245
GROMACS	Molecular Modelling	64 000	40	110
Seissol	Geophysics	64 000	31	95
waLBerla	Lattice Boltzmann	128 000	5.6	90
LAMPPS	Molecular Modelling	128 000	5.6	90
APES	CFD	64 000	6	47
BQCD	Quantum Physics	128 000	10	27

Table 1. The First Extreme Scaling Workshop results - started level was 32,000 cores (except Linpak).

However, even in light of these excellent results, it is not possible to organise these workshop with much higher than annual frequency. Finding a suitable 2.5 day time period where reserving (almost) the full capacity of the supercomputer is possible usually only few times per year. Similarly, finding dates during which all of the experts at LRZ as well as the key application developers from different projects can all travel to Munich for the duration workshop may prove to be equally difficult. So, despite its clear additional advantages (e.g. cross-pollination between different application domains), the workshop approach cannot – on its own – fulfil the full potential of a supercomputing centre as a catalyst in rapid maturing of scientific software.

VI. PARTNERSHIP INITIATIVE PICS

To address the limitations of workshop-based approach, LRZ has launched the Partnership Initiative π^{CS} (pronounced “pics”)[5] to allow more intensive and longer-term collaboration between supercomputing experts and application developers. In the π^{CS} model the supercomputing centre assigns a dedicated contact person for the application scientist, who will – during the course of the long-term relationship – take care of (among other things):

- Arranging suitable execution environments for the software
- Liaising with the software development and quality management support
- Provide training tailored to their specific needs
- Arrange access to exclusive resources, specialised infrastructure or test environments.

Ideally the dedicated contact person has a background in the application science he or she is supporting, which makes it easier to find additional synergies. In the simplest case, this benefits both internal and external communications. For example, common background makes it easy to produce press-releases and other outreach material aimed at presenting the novelty of the achievements – both in terms of IT and the basic research – to the general public. More ambitious modes of collaboration will also be more feasible, e.g. tight collaboration between the computer scientists and computational modellers will make joint publications of new joint research activities considerably easier to launch.

This approach essentially brings in one of the key approaches of mature service management (customer relationship management) as a complement to – or in some cases a replacement of – the somewhat impersonal traditional approach described in chapter IV. While this change doesn’t perhaps seem that remarkable in itself, it implicitly introduces new metric to the computing centre’s management practices: partner satisfaction and/or partner research results. Eventual monitoring and tracking of this additional metric will in turn make it much easier to present the intuitive understanding of the added value of the specific services provided. If adopted as a formally recognised metric, the data can be presented to funding agencies and other stakeholders in the centre’s governance to provide additional input for the strategic decision-making and evaluation of centre’s efficiency.

However, in the case of the leading-edge supercomputing applications, we can also assume that the customer satisfaction is also tied to the improvements in the scalability of the software. Against this backdrop, we can postulate that the additional improvements to the Extreme scaling workshop “graduate software suites” illustrate the additional improvements made possible by the partnership model. For example, the Seissol application that was scaled from 32k to 64 cores was further developed based on the partnership model. This made it possible to reach the performance of over 1400 TFLOPS using over 145 000 cores. This corresponds to 44.5% of the theoretical peak performance, which is extremely rare achievement for an actual application code.

VII. PICS IN ACTION: DRIHM PROJECT EXPERIENCE

The underlying theories and approaches behind the π^{CS} initiative were tested and refined in the final stages of the DRIHM (Distributed Research Infrastructure for Hydro-Meteorology) project [12]. The goal of the project was to create an open, fully integrated workflow platform for predicting, managing and mitigating the risks related to extreme weather phenomena. Reaching this goal required integrating numerous different components into multi-model chains that could be executed automatically to analyse e.g. flood risks (both statistical and based on specific event scenarios) and other hydrometeorological research (HMR) challenges.

DRIHM project ran from 1st September 2011 to 28th February 2015, with consortium involving partners representing major computing centres, hydrometeorological model developers and organisations with operational responsibilities (most notably Republic Hydrometeorological Service of Serbia). LRZ was involved through MNM (Munich Network Management) Team [13] that links LRZ and several academic institutions in the Munich area.

HMR has been identified as one of the key domains where improvements in the speed and accuracy of modelling would have a profound socioeconomic impact. In the 2013 paper [14] average global annual flood losses are estimated as \$6 billion in 2005, estimated to increases somewhere between \$52 billion and one trillion dollars due to socio-economic and climate changes. Thus the DRIHM project represented an application domain where socioeconomic factors clearly supported using the highest level of computing resources, even in the case where all the software components were not necessarily at the highest possible maturity level.

The starting point of the project was quite challenging: it was not only necessary to develop the software/metadata/procedural framework to link the model components together during the project lifetime, but the models themselves were in some cases used only independently (i.e. not as part of a multi-model system) in the desktop computer or small-scale computing cluster settings. The top part of the Figure 2 presents the starting point of the project, where each of the models tended to have specific demands for the execution environment and produced output files that were not compatible with each other.

The requirement gathering and design work was conducted with an approach that resembled a combination of workshop and π^{CS} models. The foundations were built in a series of project workshops or working meetings in order to ensure that the common design could accommodate all of the models (both currently identified and ones anticipated as candidates for future integration). Once these foundations were laid out, the work progressed with the participating computing centres and computer science competence centres providing named experts as contact points for the model developers and operational organisations.

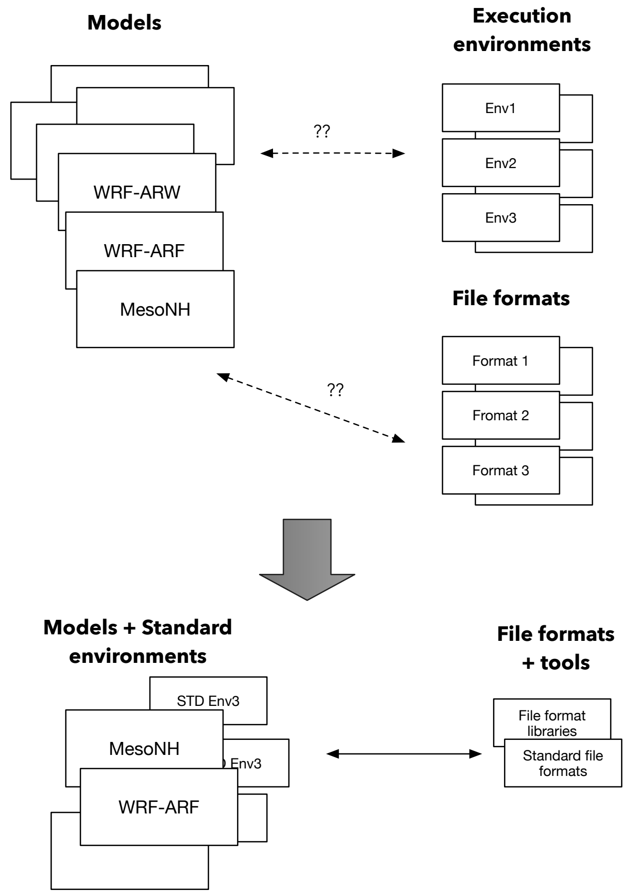

Figure 2. Modelling software framework development during the DRIHM project

In collaboration with these experts, the project developed approaches (so called M.A.P approach [15]) that made it possible to create and maintain consistently the different execution environments required by the component models as well as to perform necessary data conversion operations in a way that was verifiable in terms of syntactic and semantic compatibility (the lower part of the Figure 2). The project results validated the key assumptions behind the π^{CS} model:

- Partnering the experts together clearly reduced (in some cases eliminated) the gap between computational researchers and IT experts
- The technology and knowledge transfer was extremely efficient, allowing the project reach its goals. The automatic execution (in a matter of minutes or few hours) of a model chain that previously took weeks or even months to perform. This made it possible to consider approaches that were previously strictly limited to post-mortem analysis in operational role (early warning, steering the response etc).
- The intense interdisciplinary collaboration made a very successful "summer school" possible, which in turn triggered numerous follow-up actions.

The pairing approach also allowed a very flexible approach in terms of working meetings – during the last year several meetings scheduled on demand to address challenging issues. The success of this approach is also evident in the ability to scale some of the codes from few hundred cores to a level where running them on SuperMUC was justified.

VIII. Future directions

The future approach will be based on building on the successful application of the model in the HMR domain and extending the approach to the broader "Environmental Computing" [16] area. There are a large number of activities where problems studied are inherently interdisciplinary in nature and thus need an efficient and coherent multi-modelling approach. Expanding the number of disciplines and sub disciplines will increase the demands for the systematic collection and curation of metadata related to the model components. However, meeting this challenge is necessary in order to respond to new challenges in disaster risk reduction and other crucial activities of societies that rely on accurate modelling of natural phenomena. These activities are becoming more and more important and visible as the risk landscape is changing dramatically (due to climate change, for example). The intergovernmental response to these challenges is leading to new policies – such as Sendai declaration [17] – that give civil defence actors new mandates as well as new responsibilities.

Partnering with organisations that have sufficient visibility and credibility to drive common approach forms another strategic component in the application of the π^{CS} model. LRZ has an ongoing collaboration with United Nations Office for Disaster Risk Reduction (UNISDR) [18] in the computationally intensive tasks related to the next version of the Global Assessment Report [19]. The role of UNISDR is to collect input from different modelling activities and bring them together to produce an overview document that can be used to assess the efficiency of the disaster risk reduction policies as they have been implemented by the UN member states. Through UNISDR collaboration LRZ gains access to a very large and diverse network of experts developing models for all the different environmental disaster risk scenarios considered in the assessment report. This makes is considerably easier to identify common requirements and approaches used by the model developer community. This contact network can also be used to promote successful approaches (such as using LRZ Cloud services [20]) to a larger group of users.

The role Cloud and other virtualisation solutions will also grow more prominent, as the rapid development cycle that characterises the π^{CS} greatly benefits from the ability to tune the virtual execution environment to be as close to the original development environment as possible. This should also make it easier to adapt to the eventual situation where the "native" development environment of the environmental computing models will be the Cloud instead of the desktop. We would like to emphasise that virtualisation of the resources and HPC services that Cloud represents does not remove the challenges addressed by the two approaches presented. In fact, the "pay as you go" approach and the lack of "steps" in the capacity/price ratio where the incremental costs suddenly becomes very high (e.g. due to the need to rewire the hosting space or additional cooling requirements) may well lead to a situation where it is tempting to increase hardware resources instead of optimising the code. This will be possible up to a point, but the scalability and reliability issues will be considerably more challenging when they are (eventually) encountered. Furthermore, dealing with these issues may be more difficult due to more transient relationship with the support organisations (e.g. if the software has been run on different Cloud environments).

IX. Conclusions

Software is a crucial component in both modern scientific discovery and in providing reliable, well-designed and coherently managed IT services to support research and other human endeavours. Somewhat paradoxically, the approaches to developing these crucial components differs considerably depending on whether their role is to directly support scientific discovery or if they are aimed at providing infrastructure-level solutions for broader audience. Nevertheless, almost all of the successful infrastructure solutions have started their life – at least conceptually – as research projects, and made the (sometimes painful) transition to maturity through a fairly ad hoc process.

Supercomputing centres have a great potential to both facilitate and steer this maturing process. As the nature of their day-to-day operations incorporates aspects of both leading edge research and mature service provision, they are probably uniquely placed to act as bridge builders between the computational modelling in the basic research and software engineering and service management in more formalised services. Unfortunately the traditional approach for granting access to hardware resources – and especially to support services (such as scalability consulting and technical support) – misses quite a lot of this potential.

Based on the experiences presented in this paper, we expect the traditional approach to be complemented more and more often with partnership-oriented approaches, both as intensive, high-profile workshops and longer term partnerships. These approaches are most likely to become more and more critical as the traditional IT support - researcher relationships are challenged by the Cloud-based approach. Somewhat paradoxically we estimate that the model where the resource constraints are more efficiently hidden from the end users, the training, consulting and other services that allow researchers to use these new resources more efficiently should be offered proactively earlier in the product lifecycle than in the traditional approach to provisioning of IT services for researchers.

References

[1] GitHub service homepage, https://github.com/ (accessed 4th June 2015)

[2] SoftwareX Journal homepage, http://www.journals.elsevier.com/softwarex/ (accessed 4th June 2015)

[3] e-Infrastrucure Reflection Group White Papers, 2003 – 2013, http://e-irg.eu/white-papers (accessed 4th June 2015)

[4] e-Infrastructure Reflection Group, “Best Practices for the use of e-Infrastructures by large-scale research infrastructures”, http://e-irg.eu/documents/10920/277005/Best+Practices+for+the+use+of+e-Infrastructures+by+large-scale+research+infrastructures.pdf, p17 (accessed 4th June 2015)

[5] A. Frank, F. Jamitzky, H. Satzgera, and D. Kranzlmüller, “In Need of Partnerships - An Essay about the Collaboration between Computational Sciences and IT Services”, Procedia Computer Science, Vol 29, pp 1816 – 1824, doi:10.1016/j.procs.2014.05.166

[6] PRACE Research Infrastructure, “Resource Allocation”, http://www.prace-ri.eu/resource-allocation/ (accessed 4th June 2015)

[7] PRACE Research Infrastructure, “PRACE Preparatory Access”, http://www.prace-ri.eu/prace-preparatory-access/ (accessed 4th June 2015)

[8] M. Allalen et al, “Extreme Scaling Workshop at the LRZ”, Advances in Parallel programmin, Vol 25, pp 691 – 697, DOI 10.3233/978-1-61499-381-0-691

[9] Leibniz Supercomputing Centre, “SuperMUC Petascale System”, http://www.lrz.de/services/compute/supermuc/ (accessed 4th June 2015)

[10] Presentation: D. Kranzlmüller, “Extreme Scaling on SuperMUC”, slide 25 onward, available from: http://www.mnm-team.org/_vortraege/Kranzlmueller_20131216_Barcelona-Extreme_Scaling_on_SuperMUC.pdf (accessed 4th June 2015)

[11] Seissol working group, “High Resolution Simulation of Seismic Wave Propagation in Realistic Media with Complex Geometry”, http://seissol.geophysik.uni-muenchen.de/ (accessed 4th June 2015)

[12] DRIHM project, homepage, http://www.drihm.eu/ (accessed 4th June 2015)

[13] MNM Team, homepage, http://www.mnm-team.org/ (accessed 4th June 2015)

[14] S. Hallegatte, C. Green, R. J. Nicholls, and J. Corfee-Morlot, “Future flood losses in major coastal cities”, Nature Climate Change, Vol 3, pp 802 – 806, doi:10.1038/nclimate1979

[15] DRIHM project, “Tutorial on adapting your model to DRIHM”, Available from: http://www.drihm.eu/images/SupportCentre2014/DRIHM_MAP.pdf (accessed 4th June 2015)

[16] LMU & LRZ, “Environmental computing homepage”, http://www.envcomp.eu/ (accessed 4th June 2015)

[17] United Nations General Assembly, “Sendai Framework for Disaster Risk Reduction 2015 – 2030”, A/CONF.224/L.2, Available from: http://www.wcdrr.org/uploads/Sendai_Framework_for_Disaster_Risk_Reduction_2015-2030.pdf (accessed 4th June 2015)

[18] The United Nations Office for Disaster Risk Reduction UNISDR, homepage, http://www.unisdr.org/ (accessed 4th June 2015)

[19] UNISDR, “Global Assessment Report”, Available from: http://www.unisdr.org/we/inform/gar (accessed 4th June 2015)

[20] LRZ, “The LRZ Compute Cloud”, Available from: http://www.lrz.de/services/compute/cloud_en/ (accessed 30th August 2015)