E-Infrastructures
H2020-EINFRA-2014-2015

EINFRA-4-2014: Pan-European High Performance Computing Infrastructure and Services

PRACE-4IP

PRACE Fourth Implementation Phase Project

Grant Agreement Number: EINFRA-653838

D5.5
Application and HPC Centre Requirements for Prototyping

Final

Version: 1.0
Author(s): Michael Ott, BADW-LRZ
Date: 29.01.2016

Project and Deliverable Information Sheet

PRACE Project	**Project Ref. №: EINFRA-653838**	
	Project Title: PRACE Fourth Implementation Phase Project	
	Project Web Site: http://www.prace-project.eu	
	Deliverable ID: D5.5	
	Deliverable Nature: Report	
	Dissemination Level: PU*	**Contractual Date of Delivery:** 31 / January / 2016
		Actual Date of Delivery: 31 / January / 2016
	EC Project Officer: Leonardo Flores Añover	

* **PU** – Public

Document Control Sheet

Document	**Title: Application and HPC Centre Requirements for Prototyping**	
	ID: D5.5	
	Version: 1.0	**Status:** ***Final***
	Available at: http://www.prace-project.eu	
	Software Tool: Microsoft Word 2010	
	File(s): D5.5.docx	
Authorship	**Written by:**	Michael Ott, BADW-LRZ
	Contributors:	Jean-Philippe Nominé, CEA François Robin, CEA Carlo Cavazzoni, CINECA Olli-Pekka Lehto, CSC Eric Boyer, GENCI Dirk Pleiter, JUELICH Michael Stephan, JUELICH Andreas Johansson, LiU
	Reviewed by:	Mark Bull, EPCC Florian Berberich, JUELICH
	Approved by:	MB/TB

Document Status Sheet

Version	Date	Status	Comments
0.1	30/December/2015	Draft	First draft
0.2	07/January/2016	Draft	Modifications by FR,JPN
0.3	08/January/2016	Draft	Modifications by AJ
0.4	15/January/2016	Final Draft	Adopted comments from reviewers
1.0	29/January/2016	Final version	

Document Keywords

Keywords:	PRACE, HPC, Research Infrastructure, Prototyping Requirements

Table of Contents

Project and Deliverable Information Sheet i

Document Control Sheet i

Document Status Sheet i

Document Keywords ii

Table of Contents iv

References and Applicable Documents v

List of Acronyms and Abbreviations v

List of Project Partner Acronyms vi

Executive Summary 1

1 Introduction 2

2 HPC Centre Requirements 3

2.1 Typical Prototype Deployments 3

2.2 Requirements 4

3 Application Requirements 6

3.1 Q1 - What benefit would you expect from participating in a prototyping project? 7

3.2 Q2 - To which extent would you like to be involved in the computer architecture design process? 8

3.3 Q3 - How important is the availability of system software and tools you are familiar with? 9

3.4 Q4 - What level of technology readiness would you be willing to accept? 10

3.5 Q5 - What kind of modifications to your application would you be willing to do? 11

3.6 Q6 - Would you be willing to learn to work in a new software environment? 12

3.7 Q7 - Would you participate in a prototyping project if there was no funding for your application development? 13

3.8 Q8 - If you have previously ported an application to a new hardware platform, what exactly did you do? 14

3.9 Q9 - If you have participated in previous prototyping or co-design projects 15

3.9.1 *Q9.1 - What was the most important lesson learned?* 15

3.9.2 *Q9.2 - Was it helpful for your scientific agenda?* 16

3.9.3 *Q9.3 - What was your experience interacting with the hardware developers?* 17

3.9.4 *Q9.4 - Did the prototyping project match your expectations?* 18

3.10 Q10 - Is there anything else you consider to be important for prototyping projects and want to share with us? 19

4 Conclusions and Outlook 20

5 Annex 21

5.1 Q1 - What benefit would you expect from participating in a prototyping project? 21

5.2 Q2 - To which extent would you like to be involved in the computer architecture design process? 21

5.3 Q3 - How important is the availability of system software and tools you are familiar with? 21

5.4 Q4 - What level of technology readiness would you be willing to accept? 21

5.5 Q5 - What kind of modifications to your application would you be willing to do? 22

5.6 Q6 - Would you be willing to learn to work in a new software environment? 22

5.7 Q7 - Would you participate in a prototyping project if there was no funding for your application development? 22

5.8 Q8 - If you have previously ported an application to a new hardware platform, what exactly did you do? 23

5.9 Q9 - If you have participated in previous prototyping or co-design projects.....23
5.9.1 *Q9.1 - What was the most important lesson learned?* *23*
5.9.2 *Q9.2 - Was it helpful for your scientific agenda?* *24*
5.9.3 *Q9.3 - What was your experience interacting with the hardware developers?* *24*
5.9.4 *Q9.4 - Did the prototyping project match your expectations?* *25*
5.10 Q10 - Is there anything else you consider to be important for prototyping projects and want to share with us? 25

References and Applicable Documents

[1] http://www.prace-project.eu

[2] https://ec.europa.eu/research/participants/portal/desktop/en/opportunities/h2020/topics/715-fethpc-1-2014.html

[3] https://ec.europa.eu/programmes/horizon2020/en/news/21-new-h2020-high-performance-computing-projects

[4] http://www.deep-project.eu/

[5] https://www.montblanc-project.eu/

[6] http://www.prace-ri.eu/pcp-announcement/

[7] PRACE 1P D9.3.4 Final Report on Prototype Evaluation (19.12.2013): http://www.prace-ri.eu/IMG/pdf/d9.3.4_1ip.pdf

[8] PRACE 1IP D9.3.3 Report on prototypes evaluation (29.03.2013): http://www.prace-ri.eu/IMG/pdf/d9.3.3.pdf

[9] http://www.prace-ri.eu/prace-prototypes/

[10] http://hpc.desy.de/qpace/

[11] https://ec.europa.eu/programmes/horizon2020/en/news/eight-new-centres-excellence-computing-applications

List of Acronyms and Abbreviations

CoE	Center of Excellence
CPU	Central Processing Unit
EC	European Commission
GPU	Graphic Processing Unit
HPC	High Performance Computing; Computing at a high performance level at any given time; often used synonym with Supercomputing
PRACE	Partnership for Advanced Computing in Europe; Project Acronym
Tier-0	Denotes the apex of a conceptual pyramid of HPC systems. In this context the Supercomputing Research Infrastructure would host the Tier-0 systems; national or topical HPC centres would constitute Tier-1

List of Project Partner Acronyms

BADW-LRZ	Leibniz-Rechenzentrum der Bayerischen Akademie der Wissenschaften, Germany (3rd Party to GCS)
CEA	Commissariat à l'Energie Atomique et aux Energies Alternatives, France (3 rd Party to GENCI)
CSC	CSC Scientific Computing Ltd., Finland
CINECA	CINECA Consorzio Interuniversitario, Italy
GCS	Gauss Centre for Supercomputing e.V.
GENCI	Grand Equipement National de Calcul Intensiv, France
JUELICH	Forschungszentrum Juelich GmbH, Germany
LiU	Linkoping University, Sweden (3 rd Party to SNIC)
PRACE	Partnership for Advanced Computing in Europe aisbl, Belgium

Executive Summary

Any new technology will typically be tested and assessed in a prototype system before it is rolled out on large scale. This is also the case for HPC technologies where innovation cycles and the lifetime of deployed systems are particularly short. As part of their day-to-day business, HPC centres regularly deploy prototype systems to assess new technologies or to drive hardware development in a direction that aligns with their own requirements and the requirements of their users. Smaller prototypes are typically used to assess the features of individual components (e.g. CPUs, accelerators, memory subsystems), whereas larger systems are often deployed as part of the preparation for the procurement of next generation large-scale production systems. In the past few years, many European HPC centres have also been involved in technological research projects under different FP7 or H2020 calls that also deployed sizeable prototype systems.

Whether it is deployed within a multi-million Euro trans-European co-design project, or as part of the permanent technology assessment of HPC centres, applications are key to assess the performance of any prototype. For a small-scale and quite conservative prototype, running a few well-established benchmarks may be sufficient. But for any large-scale or very innovative prototype, it will typically require real-world applications to achieve a good understanding of the performance potential of the new hardware. To fully exploit this potential, however, the applications will have to be tuned for the characteristics of the prototype. For this kind of work, the developers of the application will have to commit to the prototyping project and do whatever it takes to adapt their code to the system.

The aim of this document is to provide an overview of the requirements of both application developers as well as HPC centres, for a hardware prototyping project. To collect the application requirements, a survey has been conducted among European HPC research projects in the FP7 and H2020 framework as well as the newly started Centres of Excellence. For the HPC centre requirements, the lessons learned from previous prototyping projects of the PRACE partners has been summarised. The focus of this document is on innovative, large-scale prototyping projects, but many of the requirements described also apply to smaller, more conservative projects.

1 Introduction

Any new technology will typically be tested and assessed in a prototype system before it is rolled out on large scale. This is also the case for HPC technologies where innovation cycles and the lifetime of deployed systems are particularly short. As part of their day-to-day business, HPC centres regularly deploy prototype systems to assess new technologies or to drive hardware development in a direction that aligns with their own requirements and the requirements of their users. Smaller prototypes are typically used to assess the features of individual components (e.g. CPUs, accelerators, memory subsystems), whereas larger systems are often deployed as part of the preparation for the procurement of next generation large-scale production systems.

In the past few years, many European HPC centres have also been involved in technological research projects that facilitated the development of new HPC technologies in Europe by means of co-design projects. These were within the FP7 framework programme, or more recently under the FETHPC call of Horizon 2020 [3]. Two of these FP7-funded co-design projects, DEEP/DEEP-ER [4] and MontBlanc [5], developed quite large hardware prototypes (each consisting of a few hundred nodes) with novel technology. This kind of effort continues within the Horizon 2020 programme that also set aside a significant budget for the co-design of HPC hardware and applications [2]. Many projects under this funding scheme will also develop prototype systems. The PRACE-3IP project is also running a Pre-Commercial Procurement pilot that should lead to the setup of two final prototypes in the 2017 timeframe [6].

Whether it is deployed within a multi-million Euro trans-European co-design project or as part of the permanent technology assessment of HPC centres, applications are key to assess the performance of any prototype. For a small-scale and quite conservative prototype, running a few well-established benchmarks may be sufficient. But for any large-scale or very innovative prototype, it will typically require real-world applications to achieve a good understanding of the performance potential of the new hardware. To fully exploit this potential, however, the applications will have to be tuned for the characteristics of the prototype. Depending on the nature of the prototype and the applications, this may be as easy as re-compiling and linking the applications with some optimized scientific libraries. Yet, it might as well require a complete re-write of some compute kernels or even large fractions of the application. For this kind of work, the developers of the application will have to commit to the prototyping project and do whatever it takes to adapt their code to the system.

The aim of this document is to provide an overview of the requirements of both, application developers as well as HPC centres, for a hardware prototyping project. Section 2 summarises the HPC centre requirements that have been collected among the PRACE partners from previous prototyping experience. Section 3 gives an overview on the requirements of application developers which have been collected through an online survey from HPC research projects in the FP7 and H2020 framework as well as the newly started Centres of Excellence. The focus of this document is on innovative, large-scale prototyping projects, but many of the requirements described also apply to smaller, more conservative projects.

2 HPC Centre Requirements

Most HPC centres regularly deploy hardware prototypes as part of their day-to-day business. They perform technology market watch as a regular task and hence know about promising new developments and how to assess new technologies. They also have expertise in running large and complex IT equipment and are used to deploy new hardware on a regular basis. They typically also have close connections to their users and know how to accommodate their needs. In some cases, they are also part of an organization involved in HPC R&D. All this makes them natural candidates to drive prototyping projects.

There is plenty of expertise in terms of prototyping among the PRACE partners in WP5. Most of them have been involved in previous PRACEP-PP, PRACE-1IP, and PRACE-2IP prototyping efforts – see for instance [7][8][9]. Many of them are project partners in FP7 and H2020 co-design projects which also develop prototype systems. All of them deploy prototype systems on a regular basis. The information provided in this chapter has been gathered from input received mainly from PRACE partners in WP5: BADW-LRZ, CEA, CINECA, CSC, GENCI, JUELICH, and LiU.

2.1 Typical Prototype Deployments

As the term “prototype” is not very specific, for the purpose of this document we consider everything to be prototype that is not a product (i.e. can be bought off-the-shelf) and distinguish between three classes of technology readiness:

- Early Development (TRL 4-5): core features available, only limited tool support, limited availability in terms of uptime, risk of wrong results.
- Late Development (TRL 6): Most features available with minor glitches and flaws, some development tools may be missing, occasional crashes and reboots of the machine.
- Quasi-Production Level (TRL 7-8): All features available, full range of development tools available, stable operation.

Obviously, the lines between each of these classes are blurry. But in general, the technology readiness grows from “Early Development” towards “Quasi-Production Level”, whereas the level of innovation typically decreases.

The size of these systems also often correlates with the technology readiness level: fancy new hardware is only bought in small quantities, whereas almost mature HPC systems that may soon be used for production may easily comprise hundreds of nodes. Unless size is crucial to assess the new technology (e.g. for scalability tests), this of course makes sense, both in terms of technology, as well as economically. Nobody is willing to spend millions of Euros on technology that may never be ready for production use. At the same time there are typically only a small number of users willing to work on these kinds of systems.

Many production-level systems that are being procured by HPC centres could also be considered as prototypes. High performance computing by its very definition is on the bleeding edge of innovation and any large-scale HPC installation is at least custom-made. Some features or components can only be tested at scale and only a few vendors have the capability to build large test installations in their labs. Additionally, some problems may only arise during typical user operation and needs to be debugged after deployment. At the very least, HPC centres normally deploy a smaller prototype system with similar architecture before they procure Tier-0 or Tier-1 system to facilitate testing and application porting. All of the above can be considered to be “Quasi-Production” systems.

As part of their ongoing efforts in terms of technology watch and their day-to-day business, HPC centres regularly procure smaller prototype systems that could be considered as “Late Development” to assess new technologies which may be interesting options for future large-scale procurements. This usually covers the likes of new CPU architectures, accelerators, memory technologies, interconnects, or new storage technology.

“Early Development” prototypes are totally different from the previously described classes. These systems usually employ novel components that have not been tested in large installations yet and may require plenty of hardware engineering and software development efforts to be realized. The intention of these kinds of prototype projects is to create impact, to drive innovation, and influence technology development instead of merely assessing and adapting technology developed elsewhere. In the past, such prototypes have often been developed within the scope of quite large nationally or European funded research projects and reached considerable scale of a few hundred to a few thousand nodes (e.g. the DEEP and MontBlanc projects, or QPACE [10]).

2.2 Requirements

As HPC centres are the entities that deploy the prototype systems, they should meet certain requirements to make the prototyping project a success. Obviously, the requirements vary with the technology readiness of the prototype itself. The more innovative the system, the more is demanded from the HPC centre.

Most requirements centre on the staff at the HPC centre and their capabilities. It requires dedicated people to drive any kind of project, but it is particularly true for prototyping projects and even more so, if they are highly innovative. Staff involved in the prototyping project need to have a good understanding of the architecture of the prototype and the underlying technologies. Often, it will require technically skilled people who are not afraid of low-level programming and/or using a soldering iron or a scope to fix problems. It is usually very time-consuming work and not something that can be handled by a system administrator on top of their day-to-day job. It also does not fit to a nine-to-five mentality as other partners involved in the project (e.g., system software or application developers) rely on functional hardware. Hence it is mandatory to be flexible and swiftly fix problems as they arise. As experience is crucial for this, it helps to have permanent staff on the payroll who have worked on prototypes before.

In terms of requirements towards the HPC centre infrastructure, flexibility and expertise is key again for all those aspects which concern the interface to the data centre. This includes, but is not limited to: power supply, cooling, networking, storage, and monitoring. As any of these requirements may change any time before or during the project, a lack of flexibility on these aspects will slow down the project unnecessarily. A clear understanding of the level of integration into the data centre infrastructure should be developed at an early point of time based on an analysis of benefits and additional efforts.

From a management point of view, it is important to define clear objectives on what should be achieved with the prototype. The objectives should ideally follow the goal of enabling something, which otherwise would not be possible, e.g. the demonstration of the usability of a particular architecture for scientific computations, or facilitating the capacity to solve certain scientific problems. Furthermore, tensions which may arise due to conflicting interests should be openly addressed at an early point in time and risk mitigation strategies need to be defined. For example, the interest of operating a possibly unstable prototype within a production

environment always conflicts with the requirement of keeping this environment as stable as possible as it comprises expensive hardware systems and services.

Hardware prototyping projects are mostly joint efforts by many parties. Although the HPC centres often drive these projects, they cannot cover all aspects and will rely on the commitment and expertise of other partners.

One of these partners is typically a technology provider or a system integrator who is responsible for the electronic, electrical, and mechanical engineering and the manufacturing of the prototype system. Similarly to the staff at the HPC centre itself, it also requires dedicated people at the technology provider to expedite the project. They need to be responsive and deeply involved in the project. Ideally, they will set up a dedicated team to work on the project with employees who can focus on it. Additionally, the technology provider should have some understanding of the environment in which the prototype is being operated.

Although system software and tools will typically be rolled out and installed by the HPC centre, it is often provided by third parties. Depending on the technology readiness of the prototype it may be available more or less off-the-shelf, or require only little adaption to the prototype. In this case, the software providers need to commit to a timeline so the software will be ready when the hardware is being deployed. For more innovative projects, such software may need to be developed in the context of the project and at some point the developers will need to have access to the actual prototype hardware. Again, the developers will have to commit to a timeline as further development downstream (i.e., applications and benchmarking) will depend on it.

3 Application Requirements

Although hardware prototyping projects are typically driven by HPC centres, applications play a crucial role in those projects as they provide the proof points for the new hardware. Only if they can be ported to the new hardware platform and run efficiently, the performance and capability of the prototype can be seriously assessed. For this, the application developers need to be dedicated, knowledgeable about their code, and have experience in performance analysis and porting of applications to new hardware.

As the persons driving the hardware side of the prototyping project typically have a different background than that of the application developers, they need to be aware of the requirements and expectations of the latter to make the project a success. Only if they have a strong incentive to commit to the project and invest working hours, the new hardware platform can be proven to be useful and the prototyping project successful.

Since the European HPC application development community is quite diverse, we decided to conduct an online survey on the requirements of application developers to get a good view on their thoughts without any bias towards certain application domains, research groups, or countries. Therefore we have distributed this survey among the application developers of current FP7 and H2020 HPC projects, as well as the new Centres of Excellence for computing applications [11]. This target group should cover a wide range of different application domains and developers with different backgrounds. They survey comprised 10 questions, most of them multiple-choice, and was conducted between November 13th 2015 and December 8th 2015 via the online platform surveymonkey.com. 28 persons responded to our request to participate in the survey. The following sections list the questions from the survey and provide an analysis of the results. The Annex in Section 5 contains the full set of responses to the survey.

3.1 Q1 - What benefit would you expect from participating in a prototyping project?

Figure 1 depicts the answers to the first question "What benefit would you expect from participating in a prototyping project?". The intention of this question was to find out about the motives of application developers to participate in prototyping projects. Three multiple-choice answers were given to choose from, more than one answer could be selected.

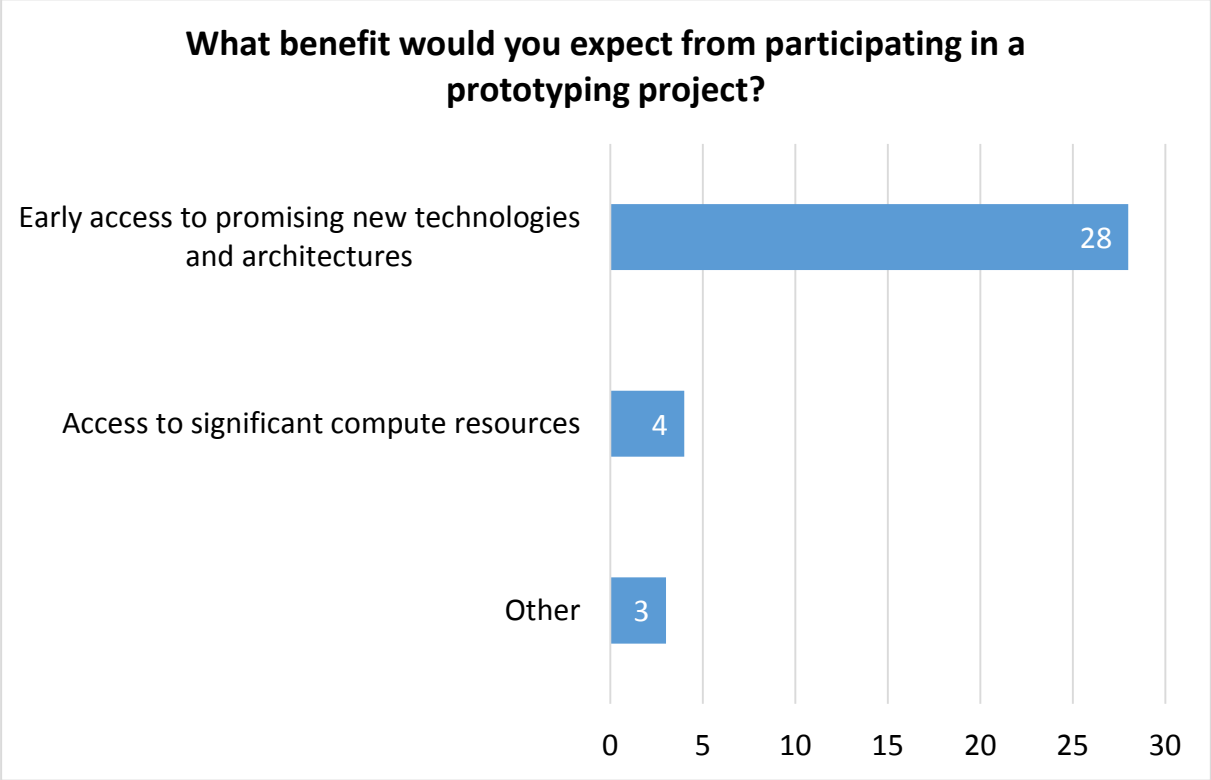

Figure 1: Answers to Q1 "What benefit would you expect from participating in a prototyping project?"

All 28 survey participants responded to the question. Answer "3. Other" also had a free-form text field to give a more detailed answer. A summary of these answers is provided in the analysis below, and the actual answers can be found in the Annex in Section 5.1.

Analysis

For all respondents, early access to new hardware is the main benefit they would expect from participating in a prototyping project. Only a small fraction (14%) hope that the prototype will provide significant compute resources for their work. The comments in the free-form text field to Answer 3 indicate that direct access to hardware developers could help their own agenda – either by influencing the hardware designers in the project to incorporate their requirements or by obtaining deeper knowledge about hardware and its efficient use.

3.2 Q2 - To which extent would you like to be involved in the computer architecture design process?

The aim of the second question was to find out to which degree application developers want to be involved in the decisions on computer hardware and design. Respondents could choose from 4 multiple choice answers, more than one answer was possible. Their answers can be found in Figure 2. The order of the answers follows the level of involvement – from no involvement to deep involvement in the hardware design process. All 28 participants answered to this question, most of them selected more than one answer.

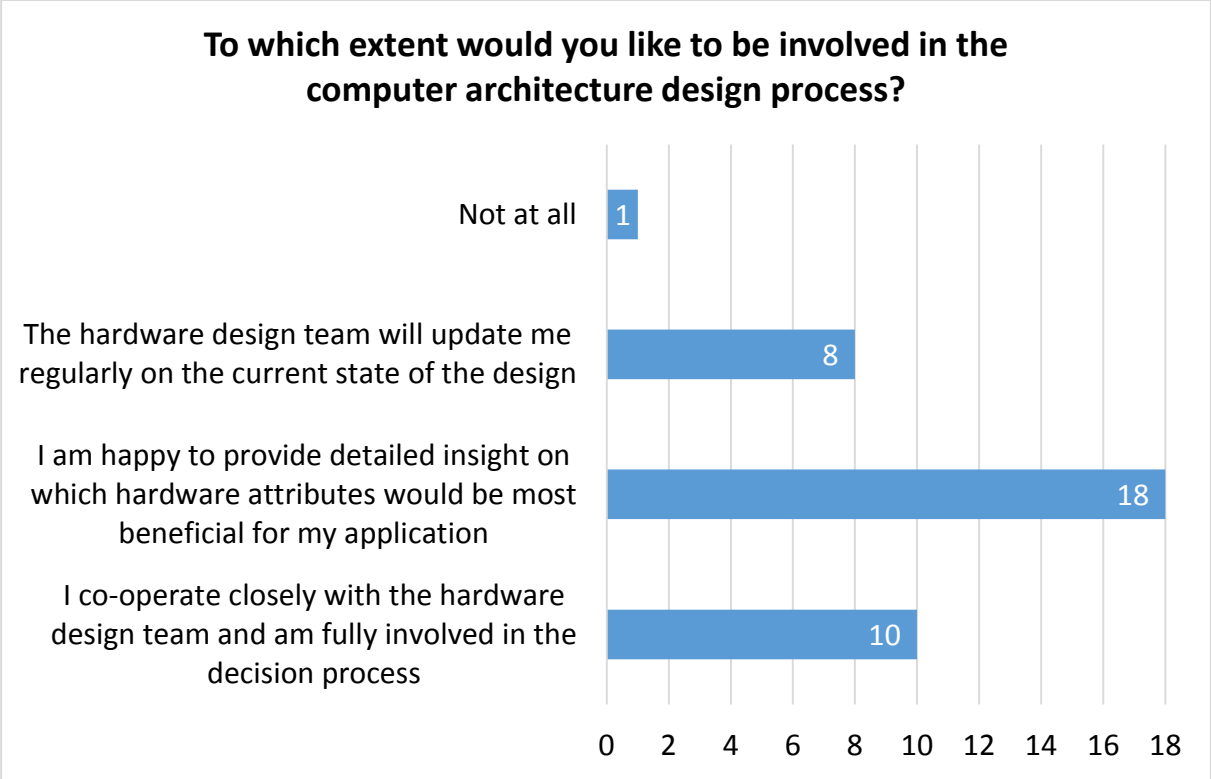

Figure 2: Answers to Q2 "To which extent would you like to be involved in the computer architecture design process?"

Analysis

Most developers want to be involved in the hardware design process. Only one didn't want to be bothered at all with any hardware details. At the very least, they want to be regularly updated on what is happening on the hardware side of the project. 64% of the survey participants (18) want to influence the hardware design decisions by telling the hardware developers what features or characteristics would be useful for their application. 10 participants (36%) even want to be fully involved and also have a say in the hardware decision process.

3.3 Q3 - How important is the availability of system software and tools you are familiar with?

Figure 3 illustrates the answers to the question "How important is the availability of system software and tools you are familiar with?". Only one answer was possible and all 28 participants answered to the question.

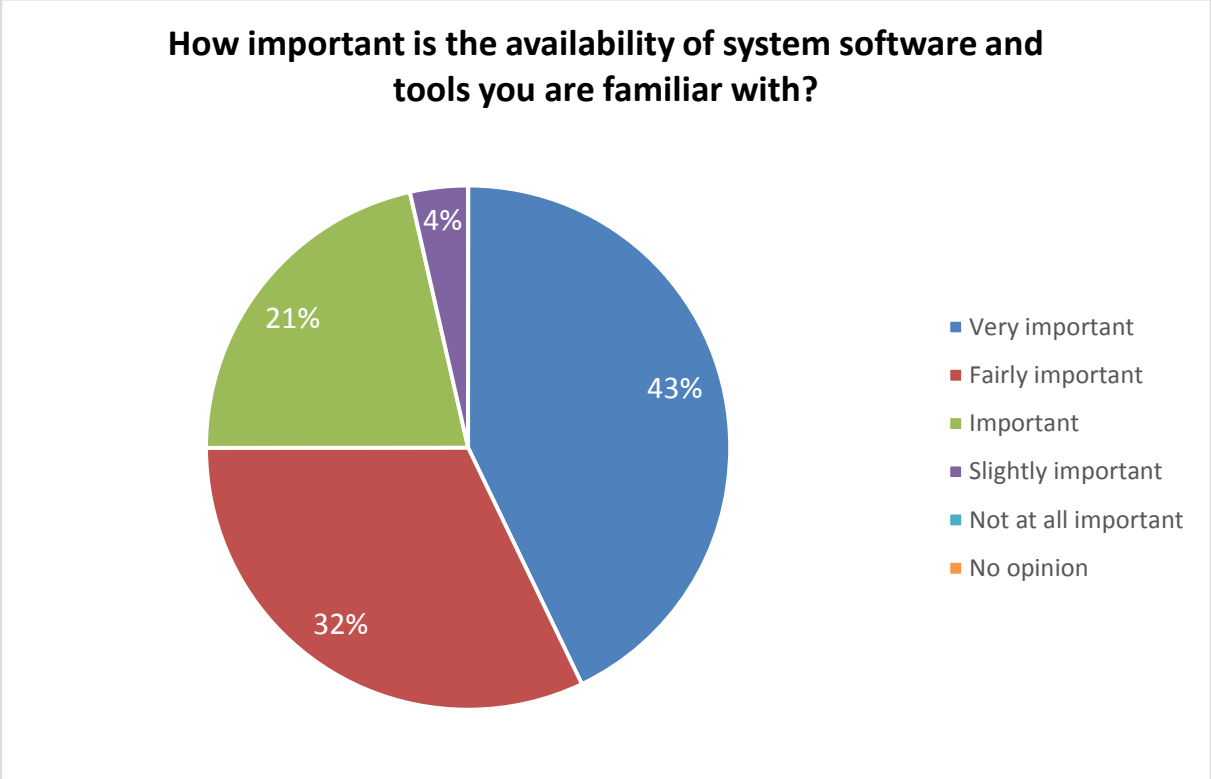

Figure 3: Answers to Q3 – "How important is the availability of system software and tools you are familiar with?"

Analysis

As can be seen from the diagram, for almost all respondents (27 / 96%) the system software and tools they are familiar with are at least important. Only for one participant it was only slightly important, whereas nobody thought it was unimportant or did not have an opinion on this. For 43% (12 respondents) it is even very important.

3.4 Q4 - What level of technology readiness would you be willing to accept?

Question 4 as depicted in Figure 4 asked for the technology readiness level would expect from the prototype system. The technology readiness levels are the same as listed in Section 2.1. Depending on the background, people may have different ideas on the maturity of a prototype system. The intent of this question was to find out whether everybody was on the same page. Only one answer to this question was possible, all 28 participants responded.

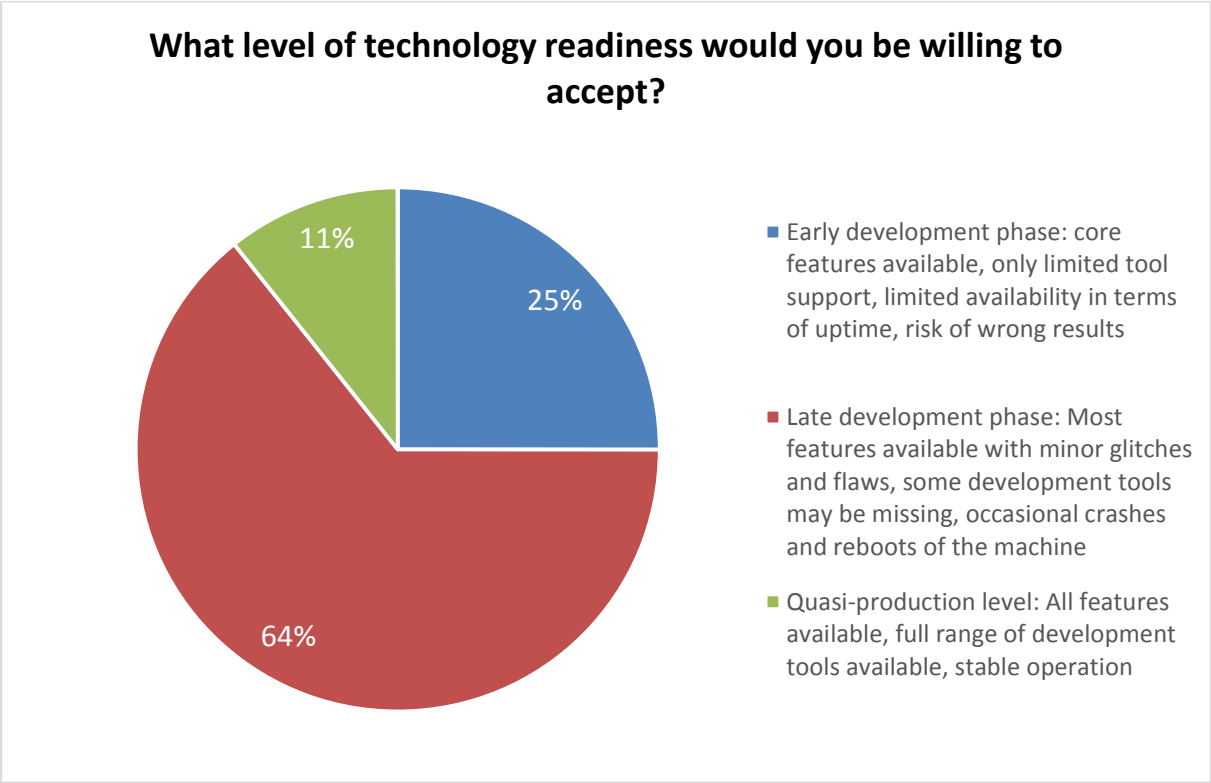

Figure 4: Answers to Q4 - "What level of technology readiness would you be willing to accept?"

Analysis

Only a quarter of the application developers (7) would be willing to deal with a very early prototype system that may crash frequently and had only limited tool support. On the other hand, only 3 participants (11%) think that a prototype system should be almost on-par with a production level system with full tool support. The majority of respondents (18 / 64%), however, would be happy with a system that may crash from time to time and had some minor flaws, as long as there was some tool support available and the machine was somehow useable.

3.5 Q5 - What kind of modifications to your application would you be willing to do?

Any new piece of hardware will typically require application developers to adapt their code to run efficiently on the new hardware. Question 5 was intended to find out how far they are willing to go to exploit the potential performance of the prototype system. As depicted in Figure 5, there were five possible answers to this question. The ordering of the answers reflects the increasing effort that is required to perform the respective modifications to an application. More than one answer was possible and the 5th answer "other" also included a free-form text field to allow for more detailed answers. All answers can be found in the Annex in Section 5.4. All 28 participants answered to the question, most picked more than just one answer.

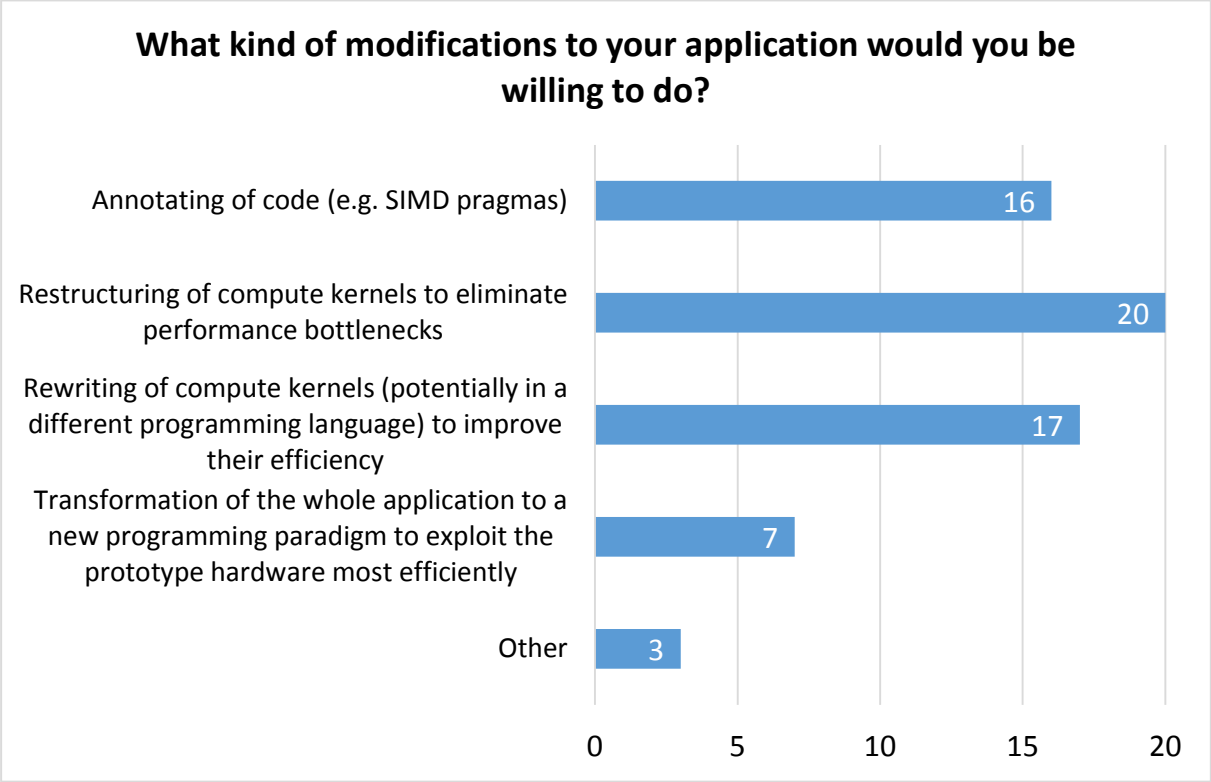

Figure 5: Answers to Q5 - "What kind of modifications to your application would you be willing to do?"

Analysis

Interestingly, there are more people willing to restructure (20 / 71%) or rewrite (17 / 61%) their compute kernels than to just annotate their code (16 / 57%). A quarter of the application developers (7), would even transform the whole application to a new programming paradigm if it helped to use the prototype most efficiently. Out of the three "other" answers, one was particularly interesting as respondent obviously felt that in an ideal world, hardware should be customised to fit the applications, not the other way around and hence they should not have to perform any changes to their application.

3.6 Q6 - Would you be willing to learn to work in a new software environment?

While the previous question assessed the willingness of application developers to make modifications to their code, this question asked how they would feel to work in a new software environment. The possible answers to this question were binary: yes or no. All 28 survey participants answered to this question.

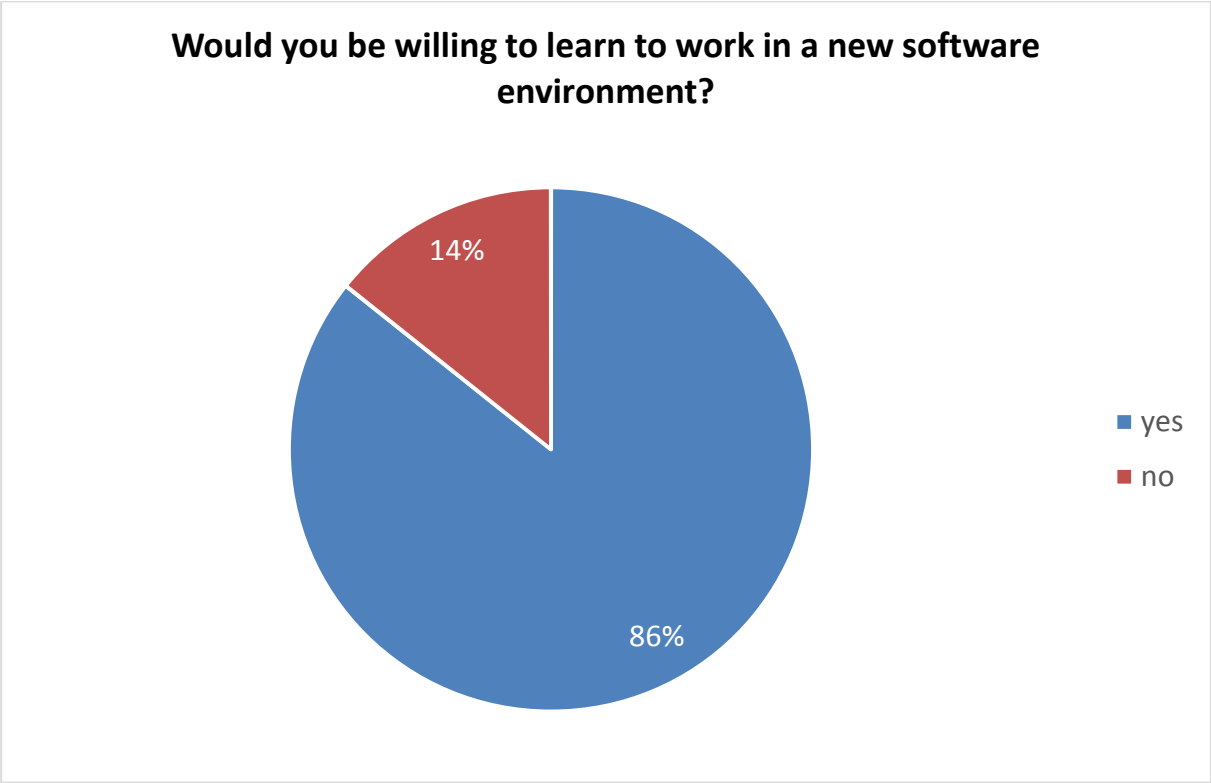

Figure 6: Answers to Q6 – "Would you be willing to learn to work in a new software environment?"

Analysis

The majority of respondents (24 / 86%) said they would have no problem learning to work in a new software environment. Only 4 (14%) felt uncomfortable with this. Obviously, these answers are somewhat contradictory to the answers given in Q3, which asked for the importance of software tools the application developers were familiar with. There, 96% said it was important to them – which is quite the opposite of the answers given here. Maybe, this question was not specific enough.

3.7 Q7 - Would you participate in a prototyping project if there was no funding for your application development?

For any project to be successful, it requires dedicated people. In a hardware prototyping project this is particularly true for the people involved in adapting the applications to the new hardware: it is their job to put it to good use. If they are lacking motivation, the whole project may easily fail. So the purpose of this question was to find out whether they have any intrinsic motivation to participate in a prototyping project and hence would also do so if they would not be funded for their contribution.
The answers to this question are shown in Figure 7. Only one answer was possible. Besides yes and no, there was a third answer "yes, if…" that allowed for a more detailed explanation in a free-form text field. Again, these detailed answers can be found in the Annex in Section 5.7.

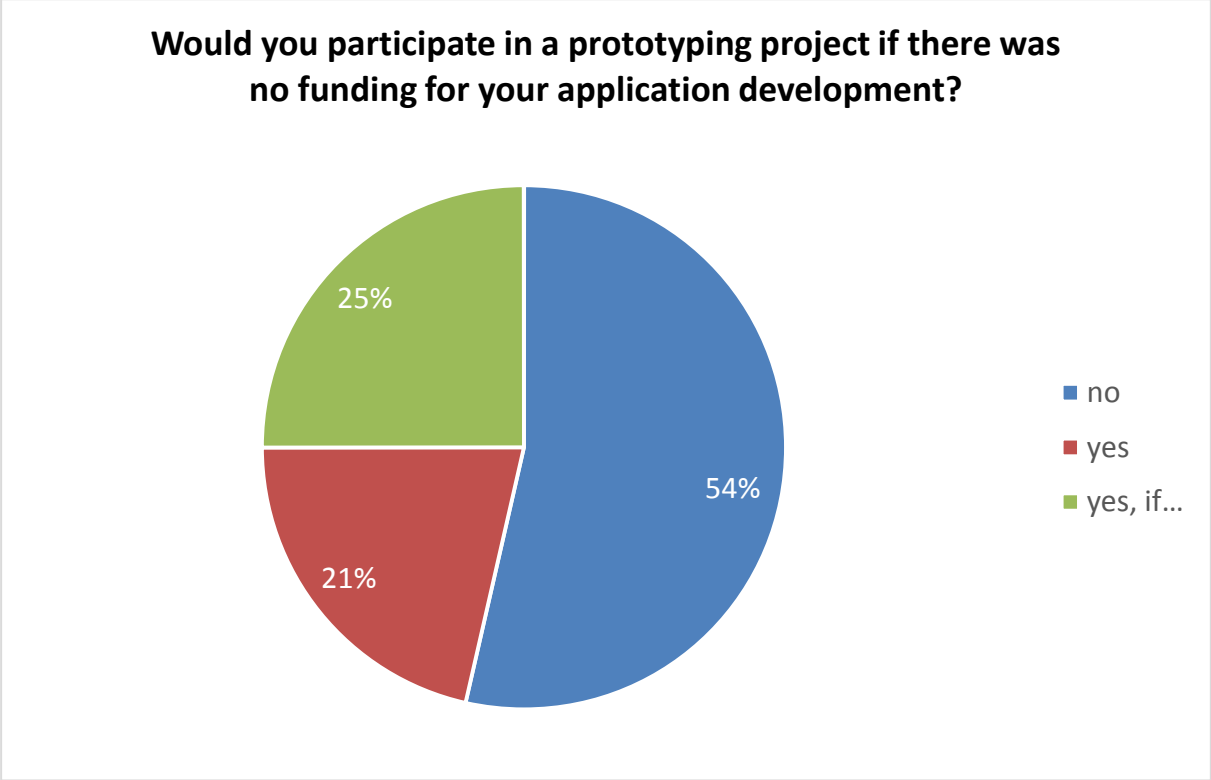

Figure 7: Answers to Q7 "Would you participate in a prototyping project if there was no funding for your application development?"

Analysis

The majority of application developers (15 / 54%) would not be interested in participating in a prototyping project if they would not be reimbursed for their effort. A quarter of them would participate even without funding, if some other condition was met. This condition was typically along the lines of "…if it was a promising project/architecture" or "…if it would be beneficial for my own work", i.e. for them the condition was that they could see some benefit for their own work in the project. Almost another quarter (6 / 21%) would unconditionally participate in such a project without any funding.

3.8 Q8 - If you have previously ported an application to a new hardware platform, what exactly did you do?

The aim of this question was to find out about the experience of developers when it comes to porting applications to new hardware and what "porting" actually means to them. Their answers are depicted in Figure 8. More than one answer could be selected, and the order of the answers reflects the increased effort required for the respective modification. Again, there was a free-form text field answer "other" that allowed for a more detailed answer. 27 survey participants answered the question, although two indicated in the "other" text field that they had no previous experience in porting applications to new hardware. The third "other" answer was that they re-structured the applications.

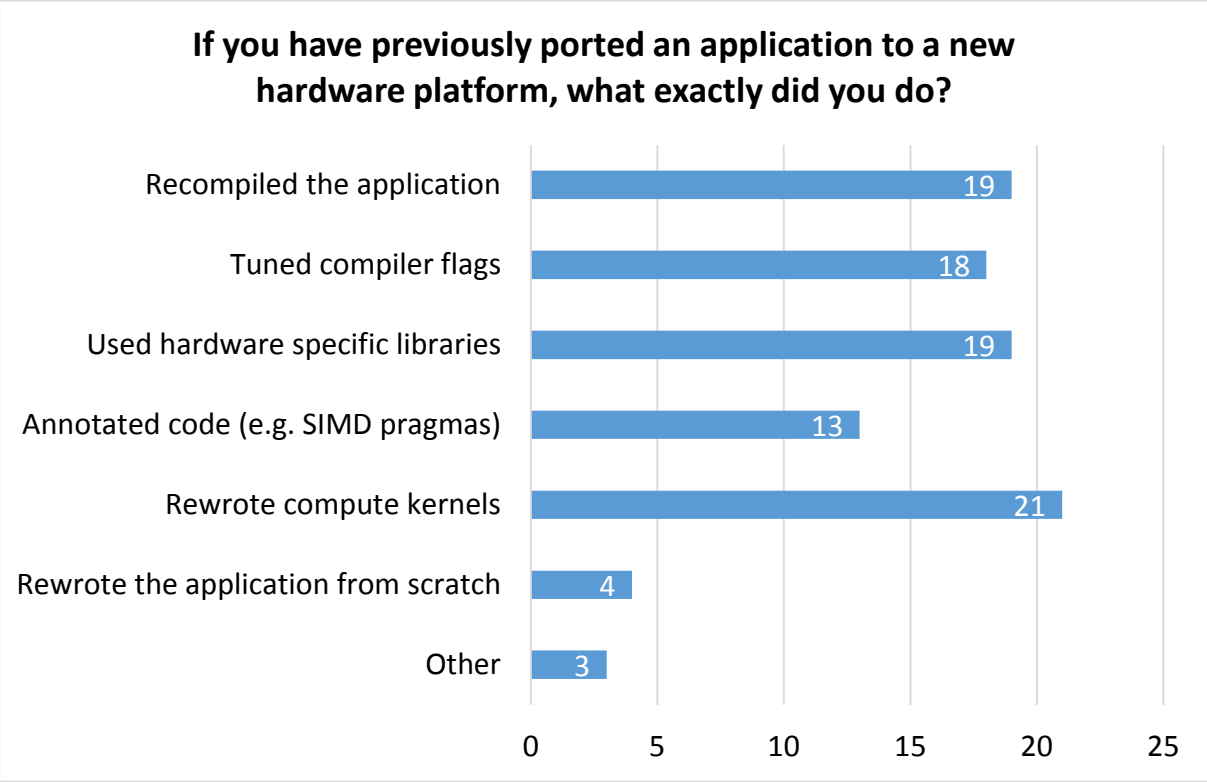

Figure 8: Answers to Q8 – "If you have previously ported an application to a new hardware platform, what exactly did you do?"

Analysis

The majority of survey participants (25 / 93%) has previous experience with porting applications to new hardware platforms. However, for three of them (12%, numbers obtained by evaluating individual answers to the question), porting stopped with recompiling the application, tuning compiler flags, and using hardware specific libraries, i.e. they did not touch the source code to adapt the application to the new platform. For all others, porting an application means also reworking the code either by rewriting some compute kernels or even the whole application. However, the latter is only an option for a minority of developers (4 / 16%). Similarly to Q5, code annotation is not particularly popular with application developers – they rather rewrote code (84% vs. 52%).

3.9 Q9 - If you have participated in previous prototyping or co-design projects

Question 9 asked for previous experience in prototyping or co-design projects. All four possible answers are free-form text fields that actually represent further questions that ask for specific aspects of the experience. Therefore, they are listed here as separate questions in the next sections. All answers can be found in the annex in Sections 5.9 to 5.9.4.

3.9.1 *Q9.1 - What was the most important lesson learned?*

As projects do not always develop as planned, it is important to be aware of potential problems and learn from the experiences others have made already. This question therefore asked for the lessons learned in previous projects.

Analysis

As all answers were free-form text, it is quite difficult to give a condensed, yet accurate reflection of all answers. However, the answers can essentially be categorized as follows:

- Properly designed/written code is essential for porting applications
- Close interaction between software and hardware developers is important
- Nothing works out of the box, everything will take longer than anticipated
- Deep knowledge of hardware and the algorithms is key
- Early access to the hardware or at least a simulator is critical

3.9.2 *Q9.2 - Was it helpful for your scientific agenda?*

Hardware prototyping projects obviously need applications as proof points for their potentially new hardware approach. But does participating in such a project also promote the scientific agenda of the application developers? To evaluate this question, all 21 answers have been categorized in 3 groups: “yes”, “to some extent”, and “no”. The distribution of answers among those categories can be seen in Figure 9.

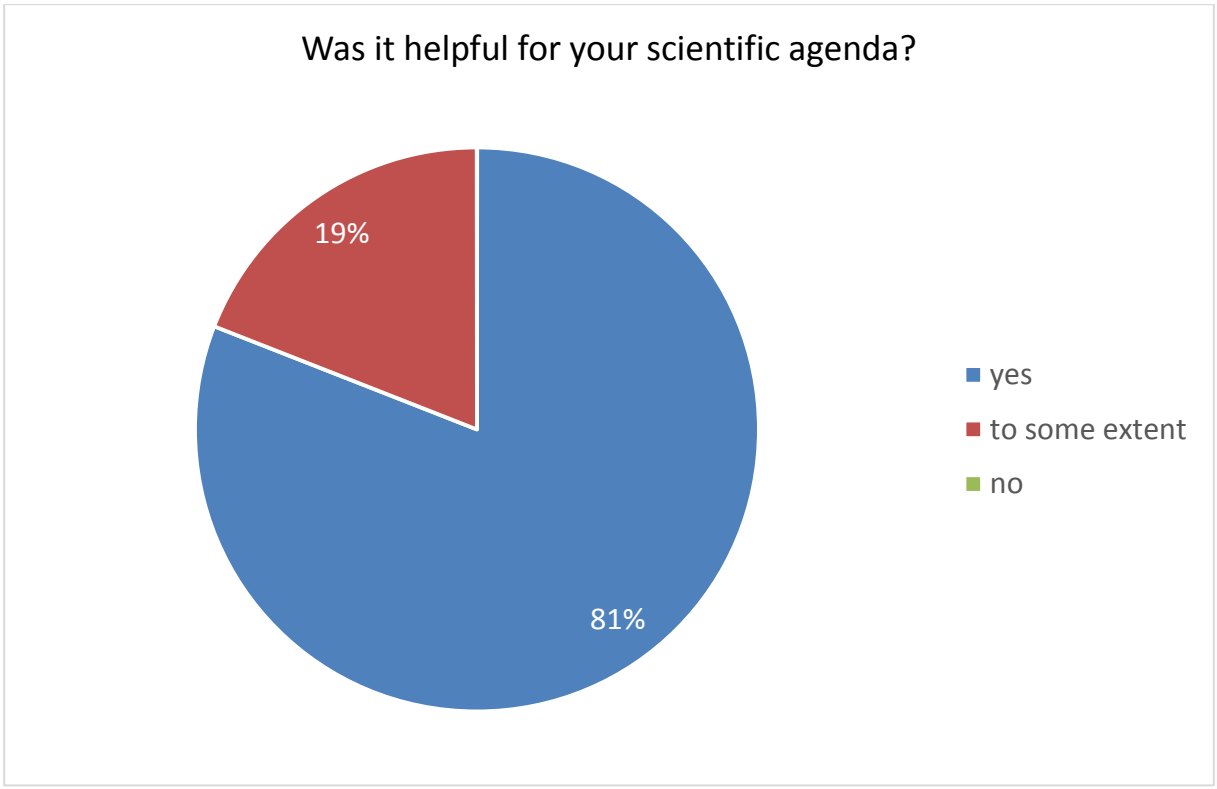

Figure 9: Answers to Q9.2 “Was it helpful for your scientific agenda?”

Analysis

All respondents found that participating in a prototyping or co-design project was helpful for their own scientific agenda. For the majority (17 / 81%) this was even true without any restrictions. Only four survey participants (19%) felt that it could have been more beneficial for them.

3.9.3 *Q9.3 - What was your experience interacting with the hardware developers?*

Since close interaction between application and hardware developers is one of the key aspects of a prototyping project, we were keen on learning about the experiences the application developers had in this regard. For the evaluation of this question, the free-form text answers have been grouped into three categories. There were 20 answers to this question, four of which have been dropped as the respondents indicated that they did not have any interactions with hardware developers. The distribution of the remaining 16 answers is depicted in Figure 10.

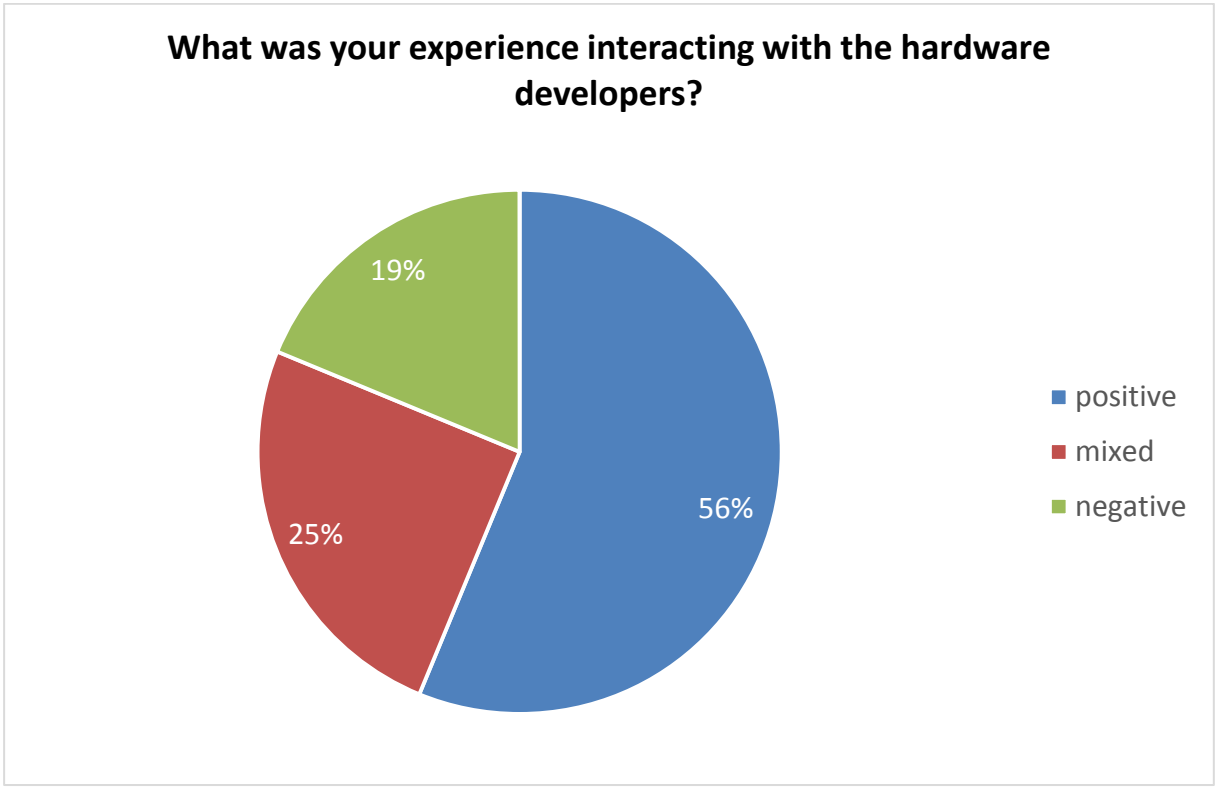

Figure 10: Answers to Q9.3 – "What was your experience interacting with the hardware developers?"

Analysis

The majority of application developers (9 / 56%) were happy with their interaction with the hardware developers. A quarter (4 respondents) felt that there was still room for improvement. Three survey participants (19%) were really unhappy with their interaction with the hardware developers. Their individual answers indicate that they were left alone and had no fruitful cooperation whatsoever.

3.9.4 *Q9.4 - Did the prototyping project match your expectations?*

The final question on previous prototyping experience asked whether in retrospect the project was what the respondents expected when they agreed to participate in the project. There were 20 answers to this question, although two indicated that their project was still ongoing and they could not say yet. The remaining 18 answers could be categorized in 4 groups that are shown in Figure 11 along with their distribution.

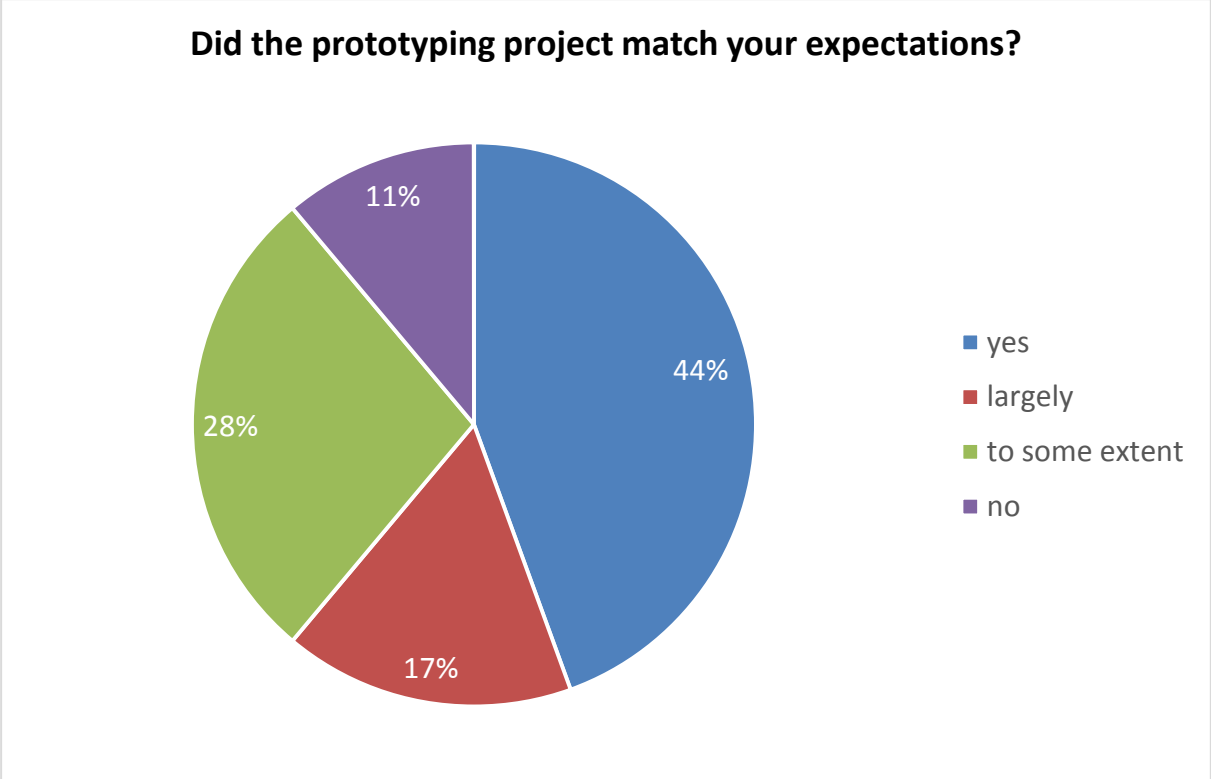

Figure 11: Answers to Q9.4 – "Did the prototyping project match your expectations?"

Analysis

For the majority (11 / 61%) of survey participants with previous prototyping experience, their respective project more or less met their expectations. For five (28%), the project was not exactly what they were hoping for when they joined in, and two (11%) expected something totally different.

3.10 Q10 - Is there anything else you consider to be important for prototyping projects and want to share with us?

The last question again allowed for a free-form text answer to provide any additional comments the survey participants wanted to share with us. There were seven comments that are provided in full text in the Annex in Section 5.10.

Analysis

The answers given to the question can be summarised as follows:

- System software and tools should be off-the-shelf - learning to work in a new environment takes time that is better spent on application development.
- Nobody can expect that large applications with hundreds of thousand lines of code will be rewritten for a single prototype system. Some sort of abstraction layer for new hardware features may help.
- If there is a fixed timeframe for the prototyping project, the hardware needs to be available early to allow for application developers to use it for their work.
- Different prototyping projects should co-operate and exchange information in order to not re-invent the wheel and duplicate work.
- The programming model is as important as hardware design.

4 Conclusions and Outlook

Prototyping projects are an important element to assess and drive the development of HPC hardware. They can be very complex any usually involve multiple parties that all need to commit to the project. Two of the most important parties are the HPC centres who typically drive the project, and the application developers who contribute the applications that are necessary to show the usefulness and capabilities of the new hardware. The previous two sections listed the requirements both parties need to meet to make the project a success, and also described the requirements they have for the project and other parties within the project.
For the HPC centres, the most important requirement is to have dedicated, capable, and experienced staff who try to advance the project. The data centre infrastructure must be sufficiently flexible to accommodate the potentially frequently changing requirements of the prototype system. Before deploying the prototype, there must be a plan on what should be achieved with the system and how it integrates into the production environment of the HPC centre.
Like anybody else, application developers need some incentive to participate in a prototyping project. For many of them, the most interesting aspect of such a prototype is to gain early access to new hardware. Nevertheless, it must also promote their own scientific agenda. Yet, if this is the case, they are typically deeply involved in such projects and many would even contribute without any financial compensation for their effort. Most of them are willing to rewrite parts of their application in order to adapt some to the hardware. However, the majority would expect at least some level of maturity from the prototype hardware and, more importantly, a certain level of tool support.
For any other party in the project, the most important requirement is again the people working on the project. They need to be resourceful, flexible, and commit to timelines.

There will be a further deliverable on prototyping in WP5 at M27: D5.6 “Best Practices for Prototype Planning and Evaluation” that will provide a comprehensive guide on prototyping activities in HPC. It will be based on the information gathered in this deliverable and the lessons learned by PRACE partners in previous prototyping projects.

5 Annex

This annex provides all answers received to the questions in the SurveyMonkey questionnaire.

5.1 Q1 - What benefit would you expect from participating in a prototyping project?

Answer	Count	Ratio
Early access to promising new technologies and architectures	28	100%
Access to significant compute resources	4	14%
Other	3	11%

Detailed answers to “other”:

1. Interact with hardware designers and provide specifications related to algorithmic peculiarities of our applications in order to get a more efficient computing system
2. Learn about aspects previously not emphasized in the code (e.g. fault-tolerance, ...).
3. Help from the manufacturer on how to best use the new technologies and architectures.

5.2 Q2 - To which extent would you like to be involved in the computer architecture design process?

Answer	Count	Ratio
Not at all	1	4%
The hardware design team will update me regularly on the current state of the design	8	29%
I am happy to provide detailed insight on which hardware attributes would be most beneficial for my application	18	64%
I co-operate closely with the hardware design team and am fully involved in the decision process	10	36%

5.3 Q3 - How important is the availability of system software and tools you are familiar with?

Answer	Count	Ratio
Very important	12	43%
Important	6	21%
Slightly important	1	4%
Not at all important	0	0%
No opinion	0	0%

5.4 Q4 - What level of technology readiness would you be willing to accept?

Answer	Count	Ratio
Early development phase: core features available, only limited tool support, limited availability in terms of uptime, risk of wrong results	7	25%
Late development phase: Most features available with minor glitches and flaws, some development tools may be missing, occasional crashes and reboots of the machine	18	64%
Quasi-production level: All features available, full range of development	3	11%

tools available, stable operation		

5.5 Q5 - What kind of modifications to your application would you be willing to do?

Answer	Count	Ratio
Annotating of code (e.g. SIMD pragmas)	16	57%
Restructuring of compute kernels to eliminate performance bottlenecks	20	71%
Rewriting of compute kernels (potentially in a different programming language) to improve their efficiency	17	61%
Transformation of the whole application to a new programming paradigm to exploit the prototype hardware most efficiently	7	25%
Other	3	11%

Detailed answers to “other”:

1. For GROMACS, data layout transformations have been the most rewarding path to taking advantages of the above kinds of application modifications, and these are also the most expensive in terms of programmer time. Such costs have to be weighed against the benefit on existing and/or conventional and/or projected hardware, and might not be considered for a prototype.
2. Ideally NONE for the intended application we are working with, i.e. NUMA scale into the multiple TB range of RAM, hundreds of cores spread over the system, fast parallel IO to fill the RAM.
3. All that I can do (which is limited due to not being the original author of the code).

5.6 Q6 - Would you be willing to learn to work in a new software environment?

Answer	Count	Ratio
yes	24	86%
no	4	14%

5.7 Q7 - Would you participate in a prototyping project if there was no funding for your application development?

Answer	Count	Ratio
no	15	54%
yes	6	21%
yes, if…	7	25%

Detailed answers to “yes, if…”:

1. The impact in the business would be clearly defined
2. if helped with my own project
3. The goals are clear. Enough time. The (internal and external) support sufficient. It is a promising project.
4. the prospect is such that it's clearly beneficial and in a sense means it's an early adoption of something that is coming anyway.
5. the technology/programming model looks promising enough to expect long-term benefits.
6. Ideally, we would prefer to participate actively in a prototyping project with funding available for application development in order to allow for a continuous co-design work. maximizing the benefits on both sides (i.e. hardware and software system

designers on one hand, and application developers on the other hand). However, a participation without funding could also be considered if we can a significant get help from the hardware and software system designers for the planned objectives on the application side.
7. The architecture and the backers of the architecture are sound enough so as to guaraty a minimum level of follow up and support

5.8 Q8 - If you have previously ported an application to a new hardware platform, what exactly did you do?

Answer	Count	Ratio
Recompiled the application	19	68
Tuned compiler flags	18	64
Used hardware specific libraries	19	68
Annotated code (e.g. SIMD pragmas)	13	46
Rewrote compute kernels	21	75
Other	3	11

Detailed answers to “other”:
1. No previous experience
2. I have not done it before.
3. Re-structured the application

5.9 Q9 - If you have participated in previous prototyping or co-design projects

5.9.1 *Q9.1 - What was the most important lesson learned?*

1. Plan to have layout of the data suit the way it will need to be used in compute kernels that run on the hardware
2. That every system needs its own programming approach
3. Have a clear defined design flow, starting from analysis of target application(s) ending with hardware verification through (almost) real applications verify
4. Good use case application, extensive tests, close interaction with the hardware developers
5. early realistic simulator needed with a sound monitoring. Stress the system to its physical limits, regardless the cost.
6. nothing works out of the box
7. To use clean and modular designs in code
8. There is still a long way to implement the technologies in the business once they are developed
9. Vendors are nervous when their prototypes don't work
10. gains are very application dependent
11. it's tedious, time consuming, and might be a wasted effort
12. The need forgood design of original code.
13. New architechtures require a lot of effort to be used efficiently.
14. Expect delays
15. Continuous feedback is important to keep hardware and application developers in the loop.
16. small and focused teams are better than large teams distributed over many institutions
17. Deep knowledge of the hardware is needed
18. News insights on algoritmic development for improved performances

19. To let hardware and software engineers talk and work together without too many presentations and management.
20. Tool support (in particular debugging) is critical.
21. access to the computing resources is important. Without it you can just extrapolate.

5.9.2 *Q9.2 - Was it helpful for your scientific agenda?*

1. Yes
2. Sure since we are also involved in hardware design
3. Yes
4. Yes
5. yes,
6. yes
7. Yes
8. Yes
9. Definitely, in particular for training PhD students
10. yes
11. to very limited amount
12. Yes.
13. The insights in the development of a HPC cluster was helpful in order to see the challenges for the next years to come regarding increased core count, etc.
14. Yes, provided ideas and funding opportunities
15. Yes, but only on the CS side of things
16. yes
17. yes
18. Yes
19. Yes, for the development of the application in both the current and future systems.
20. Yes.
21. yes. very.

5.9.3 *Q9.3 - What was your experience interacting with the hardware developers?*

1. There were no hardware developers in CRESTA co-design
2. Great since we severally interact with them in house as well
3. Good and successfull. My group is a mix of hardware/software people and application developers
4. Pleasant collaboration
5. different mentality
6. Very useful in forcing us to think about our code designs
7. It was good, they took into account our requirements
8. not applicable
9. There is a huge backend overhead with complex programs
10. was almost left alone
11. They sometimes propose technologies but without giving clear indications as to the benefits for applications.
12. The hardware developers were helpful and quickly responded to questions and suggestions
13. Very fruitful
14. Interaction was limited.
15. I developed hardware myself
16. good

17. Limited (so far)
18. Excellent. Competent and goal-focussed.
19. They have their own agenda and will not move away from it.
20. They seem to minimise the effort required for an application to fully take advantage of the architecture.

5.9.4 *Q9.4 - Did the prototyping project match your expectations?*

1. Not applicable
2. Absolutely
3. Yes
4. Yes
5. half way
6. Such work moved much more slowly than we expected
7. Yes
8. no, it was too pre-mature
9. TBD
10. not quite
11. Yes
12. In most parts.
13. For the most part yes
14. No.
15. yes
16. yes
17. Yes (so far)
18. The management process and information flow got in the way, but in many ways, yes.
19. Partly
20. somewhat

5.10 Q10 - Is there anything else you consider to be important for prototyping projects and want to share with us?

1. Compilers, toolchain and software environment should be as off-the-shelf vanilla Linux as possible. Effort invested in a prototype project carries the risk of failure, and each new thing an application developer must learn adds work that they might never re-use, and might never benefit the prototyping effort, either. Specifically, compilers should fork existing free-software HPC compilers (e.g. llvm, GNU) - my life is too short to spend any time modifying my application to deal with proprietary compiler front-end bugs and/or unfamiliar command-line flags.
2. There is, I believe, a need for a cooperation of the different prototype projects so as not to end up with developing similar applications, in the same more or less way, in several projects.
3. Any complex piece of software (>0.25M LOC) can NOT be refactored into something of temporary nature for a HOC prototype system. It should rely on the best OS support tools for insight. Hardware should provide a single addressing image, regardless the underlying limitations of the hardware.
4. There is a need to develop common platforms with device drivers already written, but not tied to proprietary hardware.
5. The prototype should be ready way before the end of the project, so that the application people have more time to actually use it within the project. Usually there exists to optimistic assumptions about the availability of hardware.

6. Questions 6 and 7 don't leave much room to answer, e.g. the willingness to learn a new software environment depends on its prospect and ease of use, so there's no general yes/no. Similarly for Q7, with no funding at all there must be some other guaranteed benefit.
7. The importance of linking innovations in hardware design with expected benefits to applications, but which do not require a complete re-write of the application.
8. Programming model is as important as hardware design.