D3.2 Best Practices in HPC Training

EUROLAB-4-HPC

Document identifier: EUROLAB4HPC-DEL-D3.2	
Deliverable lead	UNIMAN
Related Work package	WP3
Author(s)	Andrew Nisbet, Will Toms, Dionisios Pnevmatikatos, Alexey Chepstov
Contributor(s)	
Due date of deliverable	M24
Actual submission date	
Reviewed by	Manolis Katevenis, Michael O'Boyle
Approved by	
Dissemination level	Public
Website	http://eurolab4hpc.eu/
Call	H2020-FETHPC-2014
Grant agreement no.	671610
Funding scheme	CSA - Coordination and Support Action
Project start date	01/09/2015
Duration	24 months

Review record

Rev N	Description	Author	Reviewers	Date

EUROLAB-4-HPC

Contents

1. Executive Summary

This document forms the final deliverable D3.2 for WP3 "*Education*" of the EUROLAB-4-HPC project. The document outlines the proposed HPC Curriculum alongside the results of our investigation into best practices for online education. In response to comments from the reviewers at the end of the first year, the original curriculum presented in D3.1 has been updated to accommodate validation feedback on its content and structure from the HPC community. The initial best practices presented in D3.1 have been further developed as a result of our continued investigation into on-line learning, during which we interviewed educational experts as well as HPC course providers and Massively Open Online Courses (MOOC) platform representatives. These changes are outlined in detail in the next two subsections.

1.1. Validation of the HPC Curriculum

A survey was created to assess and validate the content and suitability of the proposed HPC curriculum. The survey was distributed to the members of the project consortium and to selected experts in the HPC field. We collected thirteen responses with the following main findings and modifications made to the original D3.1:

- Many responders felt that data science and analytics are a key driver of the current and future direction of HPC but not sufficiently represented in the curriculum.
- The treatment of acceleration technologies was brought up as a suggested area for improvement (e.g. such as the tensor processing unit from Google for machine learning and similar developments)
- In terms of the overall focus of the curricula, some responders found that the emphasis on parallel computer architecture was too great. When asked to propose additional courses, responders suggested advanced MPI (such as Hybrid MPI+X programming on Clusters of SMP nodes and Advanced MPI-3.1).
- Some responders also found the direct reference to Hadoop interesting but the omission of other frameworks for analytics (for example Spark) to be a significant deficiency.
- When asked to provide strong and weak points, the responders were split, with some in favour of removing parallel computer architecture and some stating that the curriculum was too focussed on software issues that we believe reflects the personal preferences and expertise of the respondents.

Consequently, in response to the questionnaire, we made the changes described below.

The emphasis on algorithms for scalable processing was increased in the overall educational goals of the curriculum. In the context of HPC and Big Data processing and analytics, "*Parallel Computing with Hadoop*" was renamed with the more general title of *"Data Parallel Computing"* that can address both present and future frameworks. The course description was altered to suggest that the most popular current frameworks (such as Hadoop and Spark) would be studied. We refocused “*Programming Heterogeneous Systems with OpenCL/CUDA*" into "*Programming Heterogeneous and Accelerated Systems*" which is more general and can address new developments in accelerators for high-performance computing (e.g. Google Tensor Processing Unit, etc). We also added a "*Data Science Fundamentals*" course to address the data-science processing explosion, and the need to be able to extract hidden values from vast amounts of operational data. Finally, in one of the suggested 2-year MSc programmes (for Physics/Math majors) we removed the “*Parallel Computer Architecture*” course and replaced it with “*Data Science Fundamentals”* to address areas for improvements in emphasis on data science.

1.2. Best Practices in HPC Training and MOOCs

The preliminary information in D3.1 provided in M12, concerning best practices in distance learning and Massively Open Online Courses (MOOCs) has been updated to reflect i) informal interviews with learning technologists and developers of MOOC course content, ii) guidance information concerning course pedagogy, content, structure, supported assessments, ECTS credits, and quality assurance procedures for the FutureLearn, and Coursera MOOCs platforms (to which full partner access was granted to the authors), and iii) the nine responses received to an online questionnaire on distance learning best practices. As a result of our investigations we found:

- xMOOCs are the most suitable course type for the deployment of a multi institution HPC curriculum such as the one described in this paper.
- The majority of European Institutions we surveyed used MOOCs predominantly as a branding tool to highlight “*beacon*” areas of research with a view to increasing the public understanding of science, and also as recruitment tool for more traditional distance learning courses.
- MOOCs are currently not used to deliver and assess degree level accredited courses, although many institutions facilitate “distance-learning” for some master’s level modules/courses.
- An individual MOOC course is approximately equivalent to 1-1.5 ECTS credits
- Individual MOOCs courses from different institutions may be chained together on the same MOOC platform to create a program/micro-masters/nanodegree.
- Creating a good MOOCs course requires a ranged of skilled professionals (in addition to the subject experts):
  - Educational professionals/learning technologists
  - Video Editors
  - Course Designers
- Course design is a very important activity that should start with the definition of measurable learning outcomes, and objectives concerning the capabilities of learners after successful completion of the course.
- High levels of social Interaction between learners (course participants) reinforces active engagement, and can help to increase MOOCs course completion rates that are traditionally very low in comparison to standard university degree and *continuing professional development (CPD)* courses.
- The role of narrative in course materials is important, and it should be maintained by ensuring sure all activities and content are closely aligned to learning objectives. Further, it is important to ensure that a narrative link is maintained and developed between course

steps and activities.
- Assessments must be carefully designed to encourage social interaction and not distract from the course narrative.
- The development of a good course is an iterative process that requires monitoring after it has been deployed and updated in response to student feedback and any delivery problems that may arise.

2. Introduction to the HPC Curriculum

Parallel and *High-Performance Computing* (HPC) is becoming more widespread, accessible, and affordable, to the extent that even general-purpose applications can now obtain benefit from parallelization (e.g. in the Big Data context). However, in order to harness the full potential of HPC infrastructures and technologies, users should be skilled and trained for HPC platforms, programming models, and tools.

This document aspires to put into perspective a wide range of requirements to the HPC learning courses and proposes a structured curriculum for HPC. Our objective in the EULab4HPC project is to establish a curriculum for technology leaders in HPC systems. Our curriculum combines courses that can be offered both in traditional and on-line forms. On-line courses can be supported as necessary by a few limited physical-presence sessions, e.g. co-located with regularly occurring EuroLab-4-HPC global events.

The initial version of the elaborated Curriculum (cf. D3.1) included the following main courses:

- Parallel Computer Architectures
- Scalable parallel algorithms
- Programming with MPI
- Parallel Computing with Hadoop
- Programming Shared Memory Parallel Systems
- Programming Multi-core and Many-core Systems
- Performance Engineering
- Programming Heterogeneous systems with OpenCL/CUDA
- Large scale Scientific Computation
- Design of HPC Clusters
- HPC Cluster management
- Advanced multiprocessor architecture
- Machine Learning
- Software Engineering
- Computational Finance
- Computational molecular biology
- Computer simulation of physical systems
- Numerical analysis and computational mathematics
- Advanced Algorithms
- Convex optimization and applications

- Particle-based methods
- Principles and applications of systems biology

The approach we followed was the following: first we solicited feedback about the desired directions of the HPC curricula from participating institutions, and from key players in HPC around Europe. We then solicited a second round of more detailed feedback on the proposed curriculum. Third, and in parallel with the target curriculum, we compiled an inventory of courses available among the participating partners to feed the best practices activity of T3.3.

We did not aim to propose a detailed, specific curriculum, but rather to provide a basis, or a framework, for the development of such curricula. It is clear that a large number of options and emphases are possible, and one curriculum cannot address all of them. Instead, we propose a flexible structure, with a suggested core set of courses along with a list of optional courses that can be extended further to address specific educational targets.

3. Educational Goals

Based on existing academic best practices, the questionnaire results (as presented in Appendix I), and additional discussions, we derive a set of educational goals for the proposed curriculum as fundamental and emergent/research-led:

3.1. Fundamental

These goals are more established, which means they are properly covered and addressed by already existing curricula:

- *Promote parallel thinking:* one of the obstacles to exploiting parallel computing is the sequential thinking that is entrenched in conventional Computer Science education. This must be overcome so that all students/practitioners develop the parallel thinking necessary for the understanding, design, development, and debugging of efficient parallel programs
- *Develop ability to program parallel systems:* Parallel programs are executed on parallel architectures and systems. The logical outcomes of parallel thinking enables the abilities of students to produce efficient parallel programs that execute correctly on parallel machines and systems, Big-Data, scale-out frameworks, etc.
- *Understanding of performance, efficiency, and overheads at* processor, compiler, runtime, communication, memory access levels. The creation of parallel programs demands the understanding (at different levels of abstraction in the hardware and software stack) of the operation of both hardware and software components. For example, in parallel program creation, this may be as ``simple'' as compiler functionality (e.g. parallelization, directives, optimizations, etc.), to exchange information (e.g. shared memory or message passing), and more complex operations of the parallel computer architecture that actually carries out computation.
- *Performance evaluation/debugging:* the creation of *correct* and *efficient* parallel programs is an iterative process that requires the identification of bugs or bottlenecks in the execution of the application and devising ways to overcome them, until the solution is satisfactory. In parallel systems this process is much more complex compared to typical sequential machines. Hence this is an important set of skills to be

acquired.
- *Knowledge of Parallel algorithms:* a vast body of work exists in the form of parallel algorithms and libraries that can be used to construct custom parallel programs. A deep knowledge of parallel algorithms is a key understanding for programmers that are required to produce efficient code across a vast range of parallel systems, languages, runtime systems, etc.

3.2. Emergent/Research-led

In contrast to the fundamental ones (Section 3.1), these goals are less established and have a moving character from research to curricula:
- *Design, programming, and use of Heterogeneous Systems:* Efficiency and heterogeneity go hand in hand, and this heterogeneity has to be integrated in the programming practices of future HPC (and in general, parallel) developers, whether it refers to programming same ISA datacentres or heterogeneous cores with GPGPUs, FPGAs, and other accelerators.
- Understanding and designing algorithms for scalable processing, in particular in the context of HPC and Big Data processing and analytics.
- Understanding and dealing with failures (and at scale).
- Understanding technology trends and limitations: power, role of heterogeneity, HPC vs. datacentre, etc.
- Understanding runtime systems and software stacks for parallel programming and scale-out parallelism: e.g. in the end, what is the difference between silk and spark?
- Understanding the changing memory subsystem (persistent and non-persistent).
- Scheduling at scale and on homogeneous or heterogeneous platforms.

We do not expect a complete program to have a course addressing each of these points, but they could be merged depending on interests/resources/etc. However, the particular research strengths and interests of a particular organisation could be highlighted using specific research led course units.

With these educational goals in mind, we worked out an extended structure for the core of the D3.1 curriculum:
- Parallel Computer Architectures
- Scalable parallel algorithms
- Programming with MPI
- Data Parallel Computing
- Programming Shared Memory Parallel Systems
- Programming Multi-core and Many-core Systems
- Performance Engineering
- Programming Heterogeneous and Accelerated Systems
- Large scale Scientific Computation
- Data Science Fundamentals

A degree of intentional overlap exists in the parallel programming courses allowing for some specialization either of the particular curriculum, or of the list of courses attended by a particular student. This set of courses covers all the basic components of any parallel system and thus provides a solid background on which specialization can be built.

More Practical

- Design of HPC Clusters
- HPC Cluster management

Application Domain Specific/Advanced Material

- Advanced multiprocessor architecture
- Machine Learning
- Software Engineering
- Computational Finance
- Computational molecular biology
- Computer simulation of physical systems
- Numerical analysis and computational mathematics
- Advanced Algorithms
- Convex optimization and applications
- Particle-based methods
- Principles and applications of systems biology

4. Suggested Courses Description

Parallel Computer Architectures

Multi-core CPUs, GPUs, desktops and datacentres, supercomputers and web sites, all rely on parallel processing to achieve higher performance. The goal of this course is to provide a deep understanding of the design, engineering, and evaluation of modern parallel computers. Beginning with an overview of technology trends the course addresses the fundamental design issues: naming, replication, synchronization, latency, overhead, and bandwidth. Using workload-driven evaluation the course gives an overview of parallel programming concepts. The course can first focus on small-scale shared memory multiprocessors and then address scalable multiprocessors and cover directory-based cache coherence, interconnection network design, software-based virtual shared memory, COMA techniques, and latency tolerance through multithreading and other means. The course will also cover the basics of programming models and communication issues in message passing and shared-memory architectures.

Programming Shared Memory Parallel Systems

Introduction to Parallel Programming concepts with emphasis on shared memory parallel systems. Shared memory overview, memory consistency, shared memory implications to programmers. Introduction to the notion of threads and the major issues in parallel programming. Programming for correctness and programming for performance. The OpenMP application-programming interface and its syntax; profile-based, incremental changes to existing serial programs. Parallel and serial regions, shared and private data, work distribution and scheduling, synchronization, reductions. Race conditions, correctness and performance considerations. Introduction of alternative Shared Memory Models: CILK, TBB, task-based parallel programming like OmpSs. Strongly suggested: hands-on, practical work in parallelizing computational kernels/small applications.

Programming Multi-Core and Many-Core Systems

The multicore chips are ubiquitous in current and will be more so in future computing systems. Their spread has transformed the software industry. To benefit from increases in hardware performance, software must be parallel. This course covers parallel programming techniques to take advantage of on-chip parallelism. The course is geared towards multi-socket, multi-core shared memory systems, touching on performance analysis and tuning, identification and management of data and task parallelism, efficient synchronization techniques for multi-core systems, efficient shared data structures, and load balancing. Strongly suggested: hands-on, practical work on multicore systems. Related: https://software.intel.com/en-us/articles/programming-for-multicore-and-many-core-products.

Performance Engineering

Large parallel systems offer abundant raw processing resources. However, exploiting this potential is a very complex process. To achieve good performance the programmer has to have a deep understanding of the underlying platform, a good insight into the computation at hand, and the skill required to create an effective mapping of the computation onto the machine. This project-based course covers topics that include performance analysis, data-structures and algorithms for high performance, cache and memory hierarchy optimization, and optimizing communication performance. For scientific computing, optimization has the form of code transformations that should be applied smartly. Several performance models of varying accuracy can guide the selection of these transformations, and expose performance and hardware bottlenecks.

Programming Heterogeneous and Accelerated Systems

The widespread appearance of GPGPUs in computer systems has already made heterogeneous computing a reality. This course serves as an introduction to the standard programming interfaces for GPGPUs, namely CUDA, and OpenCL. These programming models enable the programming of heterogeneous computers consisting of CPUs, GPUs and potentially other processing blocks as they include a way to define the platform as a host (i.e. a set of CPUs) and one or more compute devices, such as a GPU. The programmer writes task-based and data-parallel programs that can take advantage of the different types of processors in the system. Efficient utilization of GPGPUs for performance relies heavily on appropriate use of memory types (i.e. global/local, texture, constant and shared), memory bandwidth, contention, and load balancing. Therefore, efficient GPGPU programs need to orchestrate processing and data accesses, where changes to both program data structures and computational algorithms are required to generate sufficient parallelism and high performance. In addition, other types of accelerators are emerging, either in the form of reconfigurable devices (Microsoft Catapult, Intel Harp) or in a custom form (Google Tensor Processing Unit). These accelerators and their programming interfaces can also be addressed in the existing of new frameworks.

Large-Scale Scientific Computation

Scientific computing is a necessary tool in many sciences, especially in studying a wide range of physical phenomena. These applications require a very strong mathematical background, as they use well-established numerical methods and algorithms. The course can cover established algorithms and techniques for large-scale scientific computation such as iterative solvers, discrete approximation, particle methods, and similar techniques.

Data Science Fundamentals

Discover the fundamentals and the technologies that has led the data centric processing explosion, and the extraction of hidden value from vast amounts of data. Learn about the evolution of data science, the fundamental terminology and process. Cover predictive analytics, statistical concepts (regression, correlation, classification, clustering etc). Decision trees, time series, techniques for data retrieval, preparation, analysis and visualization. Large scale collection, storing and managing of data.

Scalable Parallel Algorithms

The focus of the course is the analysis, implementation, and evaluation of parallel algorithms for scalable computer architectures. It covers basic communication operations, general methods for data partitioning (e.g. blocking), performance analysis for cost and complexity. It will cover fundamental parallel approaches for numerical and non-numerical problems. Introduction to conceptual frameworks for parallelism, message passing, shared address space, PRAM. Cost models for parallel algorithms, efficiency, and scalability. Problem solving strategies: divide & conquer, pipelining, parallelization, embarrassing parallelism. Efficient communication primitives, reductions, prefix computations.

Programming with MPI

This course introduces the use of MPI to create scalable parallel programs. The course starts with the basics of parallel programming with the message passing interface (MPI) paradigm and continues on advanced programming techniques. The course covers advanced topics such as group communication, process topologies, and derived datatypes. Then it covers the debugging and performance evaluation (and performance debugging) of MPI codes. Extensions to the MPI approach (e.g. MPI+X) for improved performance and flexibility can also be addressed.

Data Parallel Computing

Large-scale data processing of massive amounts of data is a demanding task for modern data-centres ranging from system failures and non-responsive nodes to nodes of different capacity and performance. The main programming challenges addressed in this course are the partitioning and structuring of computation, and addressing fault management techniques if needed. Multiple languages, tools and frameworks can be used to process massive amounts of data. For example, Hadoop is an popular open-source framework that implements a Map-Reduce programming abstraction; it uses the Hadoop Distributed File System (HDFS) to allow for reliable distributed processing of large data sets across clusters of computers. Spark is framework for performing general data analytics on distributed computing cluster; it is a data-processing tool that operates efficiently on distributed data collections. Flink is an open-source stream processing framework for distributed, high-performing, always-available, and accurate data streaming applications.

Machine Learning and Artificial Intelligence

This course is an introduction to machine learning and statistical pattern recognition. Topics may include: supervised learning (generative, discriminative, parametric learning, neural networks, and support vector machines), unsupervised learning (clustering, dimensionality reduction, and kernel methods), reinforcement learning, and adaptive control. Applicability and requirements for big-data machine learning.

Introduction to Quantum Computing

This course is an introduction to the theory and practice of quantum computation. Topics covered may include: physics of information processing, quantum logic, qubits, superposition, entanglement, quantum algorithms, quantum simulators, quantum error correction, quantum communication, and cryptography.

Introduction to Bio-Inspired Computing

Bio-inspired computation covers approaches to computing that are inspired by nature: neural networks and cellular automata, genetic, and evolutionary computation algorithms, swarm intelligence, are examples of this approach. These systems are massively parallel and distributed in their control mechanisms. The course will cover topics such as artificial neural networks, cellular automata, simulated annealing, ant colony optimization, particle swarm optimization, genetic algorithms and other evolutionary computation systems.

5. Mapping Courses to Programmes

The proposed courses listed and described in Section 4 cover a quite broad spectrum of knowledge. However, depending on the background of the attendees, the specific educational goals, and restrictions such as the desired duration of studies, different mappings of courses to specific programs are possible and encouraged. Program duration is directly linked to the number of courses that the students should attend. Below we show how the curriculum can be adapted; we consider three program proposals with different duration and goals. First we describe two two-year programs for CS/ECE and Science (Physics/Math) majors, and then we describe a single year program that is Bologna-aligned. Similarly, other programs can be established with the desired emphasis.

2 year program, MSc level, CS/ECE majors (9 courses and semester thesis):

- Parallel Computer Architectures
- Scalable Parallel Algorithms
- Programming with MPI
- Programming Shared Memory Parallel Systems
- Performance Engineering
- Programming Heterogeneous Systems with OpenCL/CUDA
- Large-Scale Scientific Computation
- Course from the advanced/domain specific category
- Course from the advanced/domain specific category
- Hands-on Thesis

2 year program, MSc level, Physics/Math majors (9 courses and semester thesis):

- Scalable Parallel Algorithms
- Programming with MPI

- Data Science Fundamentals
- Large-Scale Scientific Computation
- Numerical Analysis and Computational Mathematics
- Performance Engineering
- Computer Simulation of Physical Systems
- Course from the advanced/domain specific category
- Course from the advanced/domain specific category
- Hands-on Thesis

1 year program, MSc level, CS/ECE majors (7 core courses, 2 elective and a two-course equivalent thesis, 60 ECTS):

- Scalable Parallel Algorithms (6 credits)
- Parallel Computer Architectures (6 credits)
- Programming Shared Memory Parallel Systems (6 credits)
- Large-Scale Scientific Computation (6 credits)
- Data Parallel Computing (6 credits)
- Performance Engineering (6 credits)
- Course from the advanced/domain specific category (6 credits)
- Course from the advanced/domain specific category (6 credits)
- Hands-on Thesis (12 credits)

Of course, the above examples can be tailored in many ways; the suggested list is just one of them. Other ways to customize the courses have to do with the exact content (and corresponding depth) of the material covered per course, and the hands-on coursework (labs, programming assignments, projects, etc). The Bologna agreement ECTS are a useful metric in gauging the level of the course at least in terms of student effort. Also, section 8 provides useful practical guidelines for detailed course development.

6. Best Practices for Online Education

In this section we present an overview of the best practices for online education and Massively Open Online Courses (MOOCs). We begin with a description of the pedagogy underpinning online education and describe how it has evolved with the development of technology. We then focus on what we believe to be the most practical online education methodology *extended MOOCs* (xMOOCs). xMOOCs are the most widely accessed online educational resource but are characterised by very low completion rates (D.F. Ohan, 2014) and learner demotivation. However, it is possible to improve these rates by incorporating a range of techniques from other distance learning pedagogies. In the next section we present best practice guidelines for transferring existing courses into successful MOOCs by exploiting a range of pedagogic principles.

History of Distance-Learning Pedagogy

Pedagogic theories for distance learning have a symbiotic relationship with the technologies used to underpin them. As technologies have advanced, pedagogic theories have evolved to exploit new features in the technology to further aid student understanding, enthusiasm and involvement. We have chosen the four main pedagogies prevalent in today's online education and define them in terms of technologies that underpinned their creation:

- Cognitive/Behaviourist (Postal Systems)
- Narrative (Multimedia/Interactive Virtual Learning Environments)
- Social Constructivist (Bulletin Boards, Email, VLEs)
- Connectivist (Social Media)

These pedagogies are discussed below and their application to Massively Open Online Courses (MOOCs) is discussed in the next section.

Cognitive/Behaviourist (CB)

The original distance learning schemes were based on postal systems, where printed material was distributed by mail. These schemes have been retrospectively called "cognitive/behaviourist" due to the prevailing pedagogies employed. In CB models of learning, knowledge was created through structured processes in which "learner's interest was stimulated, informed by both general and specific cases of overriding principles and then tested and reinforced for the acquisition of this knowledge".

Gagne's 9 general steps of instruction formed the basis for knowledge construction (Gagne, 1970):

1. Gain learner's attention
2. Inform learner of objectives
3. Stimulate recall of previous information
4. Present stimulus material
5. Provide learner guidance
6. Elicit performance
7. Provide feedback
8. Assess performance
9. Enhance transfer opportunities

Initial Behaviourist models treated the mind as a "black-box" and defined learning as a change in (externally observed) behaviour as a result of stimuli. The Cognitivism pedagogy (Ally, 2004) employed a cognitive model of the brain to facilitate the storage and retrieval of information effectively to "maximize brain efficiency and effectiveness". The structure of the brain is used to guide the construction of online learning materials in order to encourage information transfer from the sensory store to working memory, and then into long-term memory.

- *Sensory Store*: Information in the sensory store only persists for 1 second, in order to maximise the amount of information transferred to working memory it is important not to overload the senses, and to highlight critical information clearly. The reasons for learning a particular piece of information should be provided ahead of time so the learner can attend to the information correctly.
- *Working Memory*: The duration of working memory is approximately 60 seconds. Information should be chunked into pieces to facilitate processing and transferal to long-term memory. Materials should present between 5 and 9 items on a screen at any time.
- *Long-term Memory*: Information in long-term memory is stored in connected networks

that describe the relationships between nodes. Presenting information as *Information maps* and asking learners to generate such maps during the learning process aids the storage and retrieval of information to/from long-term memory. Some information that is placed in long-term memory is as a result of frequent exercise, practice and use, rather than because of a conceptual organization.

Due to the nature of the original cognitive-behavioural distance learning schemes, learning was a solitary process with little interaction between teacher and student, and zero interaction between students. Teaching presence in CB models was also reduced or reconstructed, initially to printed text then audio and video. The learning material in such pedagogical models is supposed to be self-contained and complete, requiring only teacher/learner interaction for marking and evaluation. While the initial CB pedagogy allowed for distance learning to be scaled at significantly lower cost than traditional education, the lack of social interaction reduced the ability to both construct a "community of inquiry" (Garrison, Anderson, & Archer, 2001), and/or to support networks that aid student understanding and motivation.

Narrative

A key feature of human cognition is *time sensitivity*: sensory experiences are encoded into *episodic memory* by the hippocampus in the brain, and semantic knowledge is extracted from links in episodic memory sequences. Most traditional forms of learning take the form of *narrative,* such as stories, and "the human brain makes a lot of assumptions about the causal and temporal relationship between things by their order and presentation in the narrative" (Hazel, 2008). Conventional print and electronic media has amplified the effects of narrative within learning and society as a whole.
The development of interactive multimedia and virtual learning environments, presented a host of opportunities for education via the use of different components such as video, text and interactive graphics. However, care must be taken when using such environments that the narrative of the learning process is maintained. Narrative can often be undermined by complex navigational procedures. Furthermore, a narrative is disrupted at points of interaction, such as discussion or decision points. In (Plowman, 1998), suggestions were made to preserve narrative within multimedia environments:

1. Narrative should be created specifically for interactive media, encompassing the medium and the disruptions it can generate. Keeping interaction simple will help maintain narrative dynamic
2. Tasks should be integral to the narrative. Tasks should be short discrete units that arise logically from the narrative rather than creating a diversion from it.
3. The balance between different media components requires consideration, designers should avoid an overreliance on text

In many interactive learning environments the learner has control over different routes through a programme. This can create multiple narratives, which can lead to a "multi-perspectival" set of representations of the problem. The goal of the course design is to ensure that the narratives are equally coherent.

Social-Constructivism

Social-Constructivism was a response to the lack of social aspects in traditional distance learning techniques. It was developed in conjunction with technological developments such as email and bulletin boards and other early web technologies, which allow both synchronous, and asynchronous communication, between remote parties. The pedagogy relies heavily on the theory of “Constructivism” where knowledge is “constructed” from the learner’s existing experiences rather than passively received. Some key points of constructivist learning (Koohang, Riley, Smith, & Schreurs, 2009):

- Goals and objectives are derived by the student or in negotiation with the teacher or system.
- Primary sources of data are used in order to ensure authenticity and real-world complexity.
- Collaborative and cooperative learning are favoured in order to expose the learner to alternative viewpoints.
- Assessment is authentic and interwoven with teaching.

As the name suggests, in social-constructivism, interaction between participants is fundamental to the construction of knowledge and collaborative elements of the course are as fundamental as the subject matter of the course. According to (Kanuka & Anderson, 1999) "the educator is a guide, helper, and partner where the content is secondary to the learning process; the source of knowledge lies primarily in experiences." A large amount of effort is involved in making access to online resources as easy as possible and providing a good online learning environment. A five stage online learning model, called E-Moderators, was suggested by Salmon (Salmon, 2004):

- *Stage 1*: Access and Motivation – Providing easy access to online resources
- *Stage 2*: Online Socialization – Social interaction between participants is important. This should be promoted through socially formative assignments.
- *Stage 3*: Information Exchange – E-Moderators should promote information exchange between participants by trying to engage individuals in group discussions
- *Stage 4*: Knowledge Construction – Once the collaborative learning environment has been established, knowledge construction can be achieved through discussions and group activities facilitated by the E-Moderators.
- *Stage 5*: Development – As participants become more established within the learning environment they can provide help to other participants and construct further knowledge using understanding from their own experiences.

Social-Constructivist approaches use human communications-based learning. But the focus on human interaction places limits on accessibility and results in a higher costs than the CB approach (for example to set up an accessible learning environment). It relies heavily on the ability of teachers/moderators to drive discussion and interaction and so it does not scale as well as CB approaches and may be susceptible to some of the same faults as conventional learning for example teacher domination, restrictions to geographic/temporal access.
Social-constructivism has been aided by the development of *Virtual Learning Environments (VLEs)* (Dillenbourg, Schneider, & Synteta, 2002) such as Blackboard and Moodle. In fact, Moodle cites Social-Constructivism as its defining pedagogy (Lane, 2008). VLEs are used not only to distribute course materials and results but also to provide a virtual space to facilitate social interactions between course participants.

Connectivist

Connectivist approaches (Siemens, 2005) address some of the scalability problems of the social-constructivist approach. They were developed along with networking technologies such as social networks and are based in the theory that, like social relations, knowledge is represented as a network of relationships and that evolve over time. Knowledge construction is enhanced by determining the structure and relationships inherent in new information and it's similarity to the structure of existing knowledge (patterning). Connectivist learning uses networks of interaction between participants and resources for knowledge construction. The network of resources and interaction may initially be created by teachers or instructors but their structure is determined by learners on the course, who may introduce new resources and understanding.

Social interaction is inherent to the construction of knowledge within connectivist approaches. The interactions are persistent within the network (for example in the form of comments or answers to questions from other students) and become additional resources and social capital of future networks. As interaction leads to networking between participants, more engaged students become more heavily connected within the network (more "likes", “followers” or “upvoted” posts).

Teaching presence is more complicated in connectivist approaches, and teachers merely appear as more heavily connected users in the network. Assessment is achieved within the network itself, more highly connected individuals are those who have constructed the most knowledge and have created the most social capital within the network. In this sense connectivist approaches are very similar to portals such as Stack-Overflow or the Intel Support Community, where correct or insightful responses to problems are upvoted and individual uses have a rating (e.g. black-belt) that describes how much interaction they have undertaken.

MOOCs

Massively Open Online Courses (MOOCs) refer to distance learning courses that are characterised by a large number of participants and are either free or relatively cheap to attend. The initial idea for MOOCs came out of the connectivist pedagogy when Siemens et al launched the “Connectivism and Connective Knowledge” course to describe the principles of connectivism (Siemens, 2005). The course used a range of technologies including Elluminate Inc.’s web-conferencing, RSS feeds, blogging sites and the Moodle VLE to construct a social network.

The large number of attendees in MOOCs makes employing the (staff-resource intensive) Social Constructivist pedagogy difficult to apply and therefore there are only two pedagogic styles of MOOC:

- Connectivist-MOOCs (cMoocs)
- Cognitive-Behaviourist MOOCs (xtended-Moocs or xMOOCs)

xMOOC

The overwhelming majority of MOOC courses offered today are xMOOC courses based on a cognitive/behaviourist pedagogy. The first xMOOC was the “Artificial Intelligence” course offered by Stanford University in 2011. The response to this course led to the formation of Udacity and Coursera MOOC providers. A contemporaneous course “Circuits and Electronics” offered by MIT led to the creation of the edX MOOC provider in association with Harvard. Between them the top three MOOC providers had over 24 million registered users by 2015. The majority of courses offered by the main MOOCs providers are free to attend but offer accreditation from the host institution for a small fee.

In general, most xMOOCs are either converted from existing courses or built from scratch as a series of videos covering the course content. The cognitive pedagogy may be used to structure the material to maximise cognitive efficiency. Assessment generally takes the form of a series of quizzes, which are automatically assessed in the MOOC framework. xMOOCs have been criticised by educational researchers for their lack of constructive feedback from course instructors which can lead to a low level of comprehensibility, low completion rates, and a lack of creative and original thinking amongst participants (Daniel, 2012). However, many MOOCs incorporate some form of peer-assessment into the course to improve social interaction and knowledge construction. Badgr digital badges are supported by edX that allow participants to display the courses undertaken across a range of social media platforms. MOOC providers such as Coursera are fully compatible with the Learning Tools Interoperability standards that allow a range of educational tools to be incorporated into the Coursera platform. The adoption of standard academic authentication platforms OAUTH2 and SHIBBOLETH allows institutions to setup online "campuses" for their students within the MOOC platform.

There have been many attempts to evolve the cognitive/behaviourist pedagogy of xMOOC courses into more modern constructivist or connectivist pedagogies by developing social aspects of the course. This has resulted in a range of different hybrid courses, such as Synchronous Massive Online Courses (SMOCs) (where live lectures are staged at a fixed time to increase student/teacher interaction) (Chauhan, 2014). However these courses are still experimental. It is our recommendation that the current best practice for generating an online course would be to use the existing, well established, xMOOCs frameworks and structure the course material and content using guidelines from the cognitive and social-constructivist pedagogies. In the next section we undertake a survey of the major MOOCs providers and the facilities and tools offered within their platforms.

7. MOOCs/Distance Learning Providers

In order to provide practical information within our best practice guidelines, we undertook a survey of the major MOOC platform providers. Figure 1 below shows a break-down of the different MOOCs courses by different providers. As there are too many providers to complete a full survey given the manpower resources available, we concentrate on two providers where the authors had "*partner*" access to their platforms. The two platforms considered cover more than 40% of courses offered:

- Coursera - The market leader offering the widest number of courses and active participants
- FutureLearn – A smaller UK-based provider with a strong focus on the application of online pedagogies to course content

For both providers we present a brief overview of their courses, and their business and income generation models. Finally we perform case studies on relevant HPC courses.

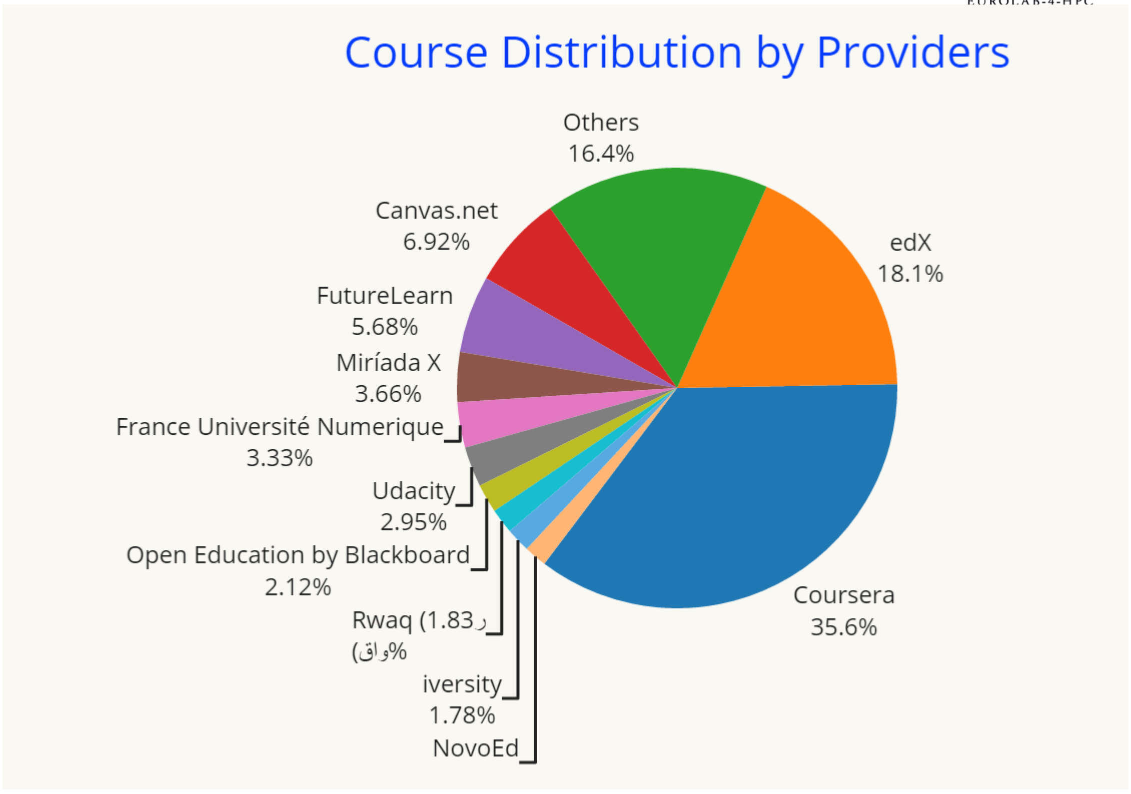


Figure 1 taken from Class Central MOOCs Report 2015

Future Learn

FutureLearn is a MOOCs/distance learning provider backed by the UK's Open University that specialises in the design, content and delivery of supported open learning courses, modules and degree programmes[1]. FutureLearn has 127 worldwide university and specialist organisation partners from Europe, Asia, South America and the USA. FutureLearn partners include the eurolab4hpc consortium members University of Manchester and University of Edinburgh, as well as PRACE. The platform launched its first courses in September 2013 and since then more than 6 million people have joined. All courses are free to join and study, apart from a newly announced paid-for course model. Upgrade payments enable additional benefits to free to join courses; such as time unlimited access to course materials, access to summative assessments/tests, and the potential to obtain certificates of completion/achievement. Certificates can be added to CVs and linked to online social media platforms such as LinkedIn. FutureLearn's business model splits revenues (after deducting operating costs associated with provisioning and running a course on their MOOCs platform) between the partner providing/developing a course and FutureLearn. Associate partners enter into agreements with FutureLearn on a course by course basis and receive a significantly reduced share of net revenue in comparison to standard partners that enter into multiple course agreements. The specific detail of revenue sharing agreements, costs incurred for operating/provisioning courses and one-off, or recurrent licensing associated with the support for the FutureLearn platform and any course content materials production are not published in this deliverable due to their commercially sensitive nature. *[Disclaimer: the information contained in this document accurately reflects the guidance and capabilities of the FutureLearn platform as of July 2017.*

[1] FernUniversitat in Hagen, Germany, is also an example of a distance learning specialist university.

The platform is under constant development, therefore the support for reusable learning objects and tools, based on SCORM, and LTI is subject to change/improvement.]

Guidelines on what makes a good course, and an online course materials are provided on the FutureLearn partner site in order to assist and guide course developers. Most of the information summarised in this section is taken from FutureLearn's help pages and their own online course that guides partners towards how to design and deliver courses that meet FutureLearn's quality assurance criteria. Course pedagogy is based on a linear narrative-led sequence of opportunities, experiences and content using constructivist learning. A key aim is to encourage active learning, for example, where learners participate in discussions and leave comments as part of completing a step or an activity. The FutureLearn platform for delivery of distance learning courses, its course creator and partner websites have been designed to make the pedagogy explicit, based on a *social-constructivist* approach. Courses are structured into *steps*, *activities* and *weeks.* The use of a narrative based pedagogy leads to practical design implications – where ideally each step needs to be linked to the one before or after by a narrative thread, and each step should have a purpose, this helps to ensures that ideas and knowledge are interconnected.

FutureLearn refers to *educators*, and *learners* rather than instructors/lecturers and students. The platform supports and focusses comment and discussion on specific steps and activities. In this way, learners have more immediate conversations on the specific educational context represented by a step rather than having a single forum associated with an entire course. Learners can follow other learners, and they receive notifications when others comment on their posts, and conversations in a manner that is similar to social media platforms such as Facebook and LinkedIn. The organisation of a course into weeks suggests a time-based progression of activities and building of ideas. Learners are responsible for marking steps and activities as complete, and dashboards give high-level overviews of progress, the percentage of the course completed and the specific completed activities.

Courses typically require 2-6 hours of effort per week, and each week is typically divided into 15-20 steps. Learners are then able to undertake course steps in small focussed chunks of time, perhaps requiring as little as 5-10 minutes of attention. In-course assessments are currently based on multiple choice quizzes, but some courses include peer review assignments that are critiqued and assessed by other learners on a unit/course. Quizzes are part of the learning process and no limit is placed on the number of retries, however tests, forming part of the assessment process have a maximum of 3 retries, and they count towards an overall course score. *Exercise steps* enable courses hosted on the FutureLearn platform to be linked to external tools and assessment methodologies using a secure https URL. *Reusable learning object* technologies, such as SCORM packages and *Learning Tools Interoperability* (LTI) specification v1.1 compliant tools can be linked into the platform as exercise steps. SCORM packages have been found to perform poorly when embedded in FutureLearn, because they often fail to meet quality assurance requirements concerning accessibility on the full range of tested devices. Mobile devices are often problematic as it is necessary to use a mobile friendly HTML5 format and to ensure that no Adobe Flash based content is still present – in order to make the content accessible for devices that do not support Flash such as those based on iOS. SCORM packages, or any other content that fail QA will not be enabled when a course runs. Typically QA must be performed at least 30days before a course is scheduled to run. The QA for courses (at the time of drafting this report in June/July 2017) performs tests on a range of devices (including MacOS 10.10, Windows10, Android5.0+, iOS10.0+) using the latest versions of HTML5 compliant browsers running Chrome, Firefox, Microsoft Edge, iOS10+ (Safari/Chrome), and Android 5.0+ (Chrome).

FutureLearn produce data-analytics on key performance indicators of the courses they run. Typically these are related to the components of a course concerning its weeks, steps and activities. Most courses on the platform will link to, and ask learners to complete, pre and post course surveys containing standardised sets of questions that enables comparisons to be made across courses and to establish benchmarks for performance. It is possible to add additional questions to the surveys via consultation with FutureLearn.

FutureLearn offers certificates of achievement that include; the learner's name, the logos of the organisations associated with a course; an outline of what was learned, and how long the course took to complete. A course upgrade must be purchased, and 90% or more of the course steps need to be marked as complete, every test question must be completed and an average of 70% must be scored across all course tests in order to be eligible for a certificate. **The typical costs of a single course certificate of achievement are 49 to 69 GBP.**

A statement of participation is a collectable printed and digital record that shows that a learner has taken part in a FutureLearn course. **The typical costs of such a statement are 19 GBP plus shipping; such statements are similar to CPD certificates that are issued after attending/completing technical training.** Under current rules, a learner has one year from the date of starting a course to qualify for a Certificate of Achievement or Statement of Participation. ID verification is required for programs that offer academic or professional credits. **Jumio** is used to perform online ID verification.

A program is typically a series of 3-6 courses designed to provide learners with a deeper and more valuable learning experience than a single course. The program may include an additional summative element, such as an exam or tutor marked assessment in order to enable learners to demonstrate that their learning progress has developed above and beyond that of the single courses. Summative assessments can be run using exams held at test centres or tutor marked assessments. The minimum and maximum study hours are expected to be 24 to 200 hours for a complete program. The costs of upgrading a program upfront may offer a discount to the learner.

Approximate costs per credit are evaluated in the context of a program/course award for 15 UK equivalent to 7.5 ECTS credits at GBP 735 (€ 830) where the example costs given in The Digital Economy accredited by the Open University, UK were GBP 236 (€ 265) for upgrading the 4 courses on the programme and GBP 499 (€ 570) to take the final assessment course, giving a total cost of GBP 735 (€ 830). The specific costs for a given number of UK/ECTS credits will be dependent on the upgrade costs for each course in a programme, and the costs associated with the final assessment. The cost of this unit equates to a cost of approximately € 110 per ETCS credit.

A collection is a number of courses from a range of partners that are linked together in order to improve search optimisation for courses related to a subject or set of skills. Collections are not designed to create a coherent learning experience – they are related courses from different providers.

A FutureLearn *award* is available to learners that have completed 90% of the steps in all courses in the program and achieved a pass rate of at least 70% on any tests. It can be used to demonstrate a deep understanding of a particular subject area and make a great addition to your CV, CPD portfolio or application for further study. **To receive a FutureLearn Award a learner must upgrade all the courses in a program and qualify for a Certificate of Achievement on them all**

FutureLearn offer postgraduate degree qualifications – graduate certificate (4 programs), graduate diploma (8 programs), master and master of professional practice (16 programs). Currently, they are internationally accredited by Deakin University, Australia. The total cost of acquiring a Masters postgraduate degree in Professional Practice: Information Technology from Deakin is a total of AUD 18090 (€ 12255), with a breakdown of 4 Deakin credentials of professional practice (AUD 495 each), + 2 FutureLearn Programs (AUD 3285 each) + Deakin Capstone (AUD 6570). A capstone project is typically a significant assessment where learners must demonstrate the application of knowledge learned and apply or examine the knowledge in the context of a specific project or idea. **The cost of this masters programme equates to a cost of approximately €200 per ECTS if we assume that it is equivalent to a 60 ECTS Masters programme. The University of Manchester fee cost for an MSc in Advanced Computer Science ranges from GBP 10000 (UK/EU) to GBP 21500 (international) and this equates to a cost of € 190 to € 400 per ECTS credit.**

The learning design phase of course development typically starts with defining learning outcomes, having verb based outcomes and meaningful competencies and activities that can be assessed. The content of what will be used to teach a course, such as videos and articles needs to be defined, as well as what planned meaningful interactions will be used to enable learners to meet the learning objectives. It is important to understand who the learners are, and their backgrounds or personas

A short video should be used to introduce a course. Videos should typically be less than 8 minutes, be downloadable, and have subtitles and/or transcripts. A course should begin by forcing student to complete/enter a comment on a discussion board. Moderators review and restrict content in comments according to FutureLearn's code of conduct conditions. Learners should be prompted both to comment on discussion boards, and to mark any steps in a week as complete. This is to reinforce the goals of making learners engage in a course, and to celebrate completion of tasks. Learners should be challenged to reflect on what they've learnt, and invited to reflect on how a course relates to their own personal experiences or goals in taking a course. Comments need to be more than just simple *yes* or *no* answers in order to inspire learning experiences centred on storytelling/narratives – that provoke conversation, and that celebrate progress – by marking steps as complete. Celebrating progress relies on the concept of step completion, where learners are responsible for taking the action to indicate that have reviewed content and engaged in discussions/tests. Quizzes/tests are used to give learners an opportunity to check their understanding of concepts/ideas.

FutureLearn Course Case Studies

- Supercomputing

  The course is a 5 week course requiring 3 hours per week of study, it is designed for anyone interested in leading-edge computing technology, supercomputers or the role that computer simulation takes in modern science and engineering. All technical aspects are covered at a conceptual level and there is no requirement to be able to write computer programs. However, anyone with existing programming experience will learn how programming modern supercomputers differs from programming a home PC. A course upgrade is available for GBP 59. The course is at an introductory level, and would seem to be targeting the public understanding of science and how supercomputing enables the simulation of physical systems rather than developing technical knowledge in high performance computing.

- Managing Big Data with R and Hadoop

  The course covers basic functionality of the statistical programming language R, Apache Hadoop and RHadoop. It is designed for advanced final year undergraduate and first year PhD postgraduate researchers interested in data science, computational statistics and machine learning that need to understand how to exploit HPC resources. The course would appear to be an example of a university research student support course lasting 2 days that has been rolled out into a distance learning course. The duration of the course is 5 weeks with 4 hours per week, and an upgrade is available for GBP 69 that enables unlimited access to the course, access to tests and the potential to obtain a certificate of achievement.

Coursera

Coursera was founded in 2012 by two Stanford Computer Science professors, Daphne Koller and Andrew Ng in order to be able to allow university courses to be shared online for anyone to take. The Coursera platform has since grown to have more than 25 million learners, 149 university partners, 2000 courses, 180 specializations (collections of courses) and 4 degrees. Coursera, like FutureLearn, also recommends a learning objectives based approach to course design, referred to as backwards design – where the end goals of a course are defined first. The end goals are the future state of learners after they have finished a course. The learning objectives are defined, and then assessment activities are defined. If it is not possible to define the summative assessments then the learning objectives must be revisited and revised. Once the summative assessments are defined then the design of the instructional materials can progress. It is important that the learning objectives and any levels of competency or proficiency in any skills or tasks that learners are expected to achieve can be measured and evaluated by the summative assessments.
Coursera, like FutureLearn, recommends a modular approach to learning, with at least 4 measurable objectives, 15 hours content delivered in 4-6, with each week split into modules that form a series of lessons/learning activities. Coursera recommends that each lesson should be completed in 30-45 minutes. Each module should have at least one summative assessment apart from the initial course welcome module. Summative assessments should reflect learning objectives at higher levels of Bloom's taxonomy concerning analysis, synthesis and application that require more complex and in depth learner engagement and typically require more than 30 minutes to complete.
The minimum requirements for a specialisation, is that there are 4-7 connected courses, having clear learning objectives with a clear intended audience. Each course should have a project, and it is typical to have a culminating project across the specialisation that is its own course with the goal being that a learner can demonstrate the levels of assimilated knowledge that they have synthesized across the specialization. Such culminating projects are often referred to as capstone projects. There should be typically 20-40 hours of learner engagement in projects across all the courses in a specialization. For courses in Data Science and Computer Science topic areas, it is recommended to include at least one assessment that requires *programming or coding*.

Coursera Programming Assignments

Coursera directly supports programming formative and summative assignments via in-browser *codeblocks* and *Jupyter notebooks*. This is one area where the Coursera platform is more advanced than FutureLearn. The benefits of in-browser tools are that no setup is required by the learner, and no jumping between the course and the input screen windows. Codeblocks are interactive self-contained coding environments that are embedded into course content, such as quizzes, in-video questions, and readings. Learners can input text into an in-browser code editor, "run" the code, and see the output. The main programming languages supported by codeblocks and Jupyter notebooks are listed on the Coursera partner help site here. Jupyter notebooks can also provide additional languages using the kernel support listed here. A different list of interesting computer science and related programming courses making use of Jupyter notebooks is provided at this github link. Codeblocks are useful for introducing basic concepts such as basic language syntax for declaring and assigning to variables, as well as basic control flow if/then/else and loop-based constructs for iteration and repetition. A Jupyter notebook is an open-source web application that allows the creation and sharing of documents that contain live code, equations, visualizations and explanatory text. Jupyter notebooks have been integrated with big-data and data-science programming frameworks and languages such as Spark, R and Scala. Built-in grading tools are available to check output using numeric and regular expression output. Custom grading can be accomplished by building and providing a Docker image for a self-contained linux image that can run and grade an assignment. Coursera provides the infrastructure to invoke graders in a secure, scalable, and timely manner. The custom grader's sole responsibility is to evaluate a single part of a single submission, and to produce a score and output that is then shown to learners. An example github link demonstrates the main features of a custom grader.

edX Course Case Studies

- Chalmers Computer System Design Courses

Relatively advanced final year undergraduate and postgraduate courses entitled *Computer Systems Design for Energy Efficiency* and *Computer Systems Design: Advanced Concepts of Modern Microprocessors* have been adapted into MOOCs offerings on the edX platform. The courses last for 4 weeks, having a graded summative assignment each week, a collection of short video material with talking head shots followed by quizzes. The quizzes do not allow full-text programming, but the selection and ordering of predefined assembly programming statements does allow a reasonable level of testing the attained competency and understanding of how advanced microprocessors operate. The videos are focussed on specific concepts, use transcripts and are downloadable. In some videos, dynamically animated word clouds are used to help the learner assimilate information from the video. The word clouds and any text in the videos, and associated screencasts of Powerpoint use suitably large fonts. As a social and learner community building exercise, during the initial first week, learners are encouraged to place their approximate worldwide location on a map associated with the course.

- Harvard’s CS50X Course[2]

Harvard’s normal CS50 Introduction to Computer Science is roughly entitled as an *Introduction to the Intellectual Enterprises of Computer Science and the Art of Programming.* CS50X is the edX version of the CS50 course that can be taken without enrolling directly at Harvard. The course is unusual for a MOOCs course in that it attempts to largely replicate and enhance the content of the on-campus CS50 course. The course is structured into 11 weeks (including the introductory week), it has 9 problem sets, and a final project that are expected to take 10-20 hours of effort each. Learners can obtain a verified certificate if they pay 90 US$ and they attain 70% or more on each and every

[2] The information in this section comes from examining the edX pages for the course, and from an interview with David J Malan the main developer and instructor of CS50/CS50X.

problem set including the final project. Events such as puzzle days, focussed on problem solving, and hackathons are organised that can be attended both in-person and virtually. Different social media platform views and support tools for CS50X are provided for Facebook, Twitter, Instagram and Snapchat in order to encourage and foster a community of learners. A web-based integrated development environment IDE is provided that enables learners to program in the cloud without installing any software locally. The IDE uses a version of Ubuntu linux that has been containerised with Docker in order to provide clean and simple programming environments. The IDE environment provides users with 512RAM in which to run programs and 5G of cloud disk storage in which to save their own files and directories. Users do not need to install any software locally. CS50X is unusual in that it goes against general wisdom and advice for MOOCs, as its content is not broken up into small chunks. CS50X is the delivery of a traditional CS course that has been greatly augmented with extra support and learner interactions in order to facilitate online distance learning. An interesting aspect of the CS50X course is that Launchcode uses a version of the course as a means to provide educational outreach with the goal of providing students with the foundational knowledge required to obtain employment in the technology industries via apprenticeships. In this way, Launchcode provide added value to CS50X that enables disadvantaged learners to complete the course and then for the learners to obtain a route into employment and further study options.

Summary of Coursera and FutureLearn Approaches

All information concerning the business models of MOOCs providers in this document was taken from information in the public domain. MOOCs business models are based on revenue sharing of learner fees paid for access to course materials and assessments, and certificates of course completion and achievement. Course fees are typically paid on a piecemeal basis, otherwise free access is typically time limited, and summative assessments cannot be accessed without payment. Pricing discounts may be available if payment for access to a series of linked courses is made in one transaction. Courses that are linked or related to a specific area or problem domain are commonly referred to by various MOOCs providers as a *program* (FutureLearn), *specialization* (Coursera), *micromasters (edX)* or a *nanodegree* (Udacity). Any certificates or credentials associated with achievement on successfully completing a course or specialization may be recognised or endorsed by employers, and even accepted as academic credit for entry into a degree program. Endorsement, or partnership of courses with employers is generally likely to make a course more popular and can assist with marketing courses to prospective students.
It is the opinion of the authors of this document that continuing professional development courses related to HPC that do not require rigorous summative assessment are low risk candidates for testing the early development, and delivery of MOOCs courses. Candidate courses might include those currently developed and delivered by university research support departments, and training centres that aim to help learners such as new research students and staff to exploit and optimise their programs and applications with basic and intermediate levels of expertise on HPC and parallel computing resources, potential example courses could include those dealing with CUDA/OpenCL, OpenMP, MPI, and more ambitiously those concerning application domain specific physical simulation frameworks for example, in application domains such as computational molecular biology (bioinformatics).

Table 1 contains a comparison of Coursera and FutureLearn platform and course guideline information where there are small and/or significant differences in their approaches. Table 2 summarises common information to Coursera and FutureLearn platform and course content guidelines. It additionally contains information concerning the approximate costs per ECTS credit, and how to relate ECTS credits to other academic schemes such as used in the UK and the US.

Table 1 Comparison of Coursers and FutureLearn features and guidelines.

Criteria	FutureLearn MOOCs Platform	Coursera MOOCs Platform
Platform approach to tackling learner plagiarism	Tutor or teaching assistant marked assessments, and/or exercise steps can be linked to the turnitin tool used by many universities that has support for plagiarism detection in text documents.	Coursera honour code prohibits plagiarism no specific or special tools to prevent plagiarism are inherent in the platform itself. Coursera can link to turnitin tools.
Course effort – similar approaches are taken in both platforms	2-6 hours per week per course, 15-20 hours total, and weeks are divided into 6-20 steps. Aim is to allow learners to take course in chunks that may be as small as 5-10minutes of focussed engagement at a time.	At least 15 hours content delivered in 4-6 weeks, with each week split into modules that form a series of lessons/learning activities. Each lesson should be completed in a maximum of 30-45 minutes.
Assessment guidelines – similar approaches are taken in both platforms	Guidelines on the inclusion of formative and summative assessment are given on the partner website. Less prescriptive than Coursera. Balanced usage of different assessment types are recommended across steps and weeks. Assessments should be related to learning objectives and transferable skills and competencies that a course seeks learners to achieve.	At least one summative assessment (apart from the course welcome) and one formative assessment per module. Use explanatory feedback where feasible. Relate assessments to learning objectives and vary the use of questions, discussion points and quizzes across modules.
Pricing costs for upgrading a course or specialization for assessment or for a certificate of completion or participation.	FutureLearn typically charges 49-69GBP (approximately €55-80) per course, this entitles a learner to statement of completion if 90% or more of all course steps are marks as complete. Certificate of achievement requires an assignment/test score average of 70% or greater. Optional online identity verification is possible to add an ID verified badge to a learner's profile and a digital/printed certificate. Free course access is typically time limited to 14 days, and	Coursera requires learners to have completed photo ID verification using a currently valid and accepted ID such as driver's licence, passport. Price of course certification is typically US$39-79 (€35-70) for a course. By default all courses offer a course certificate on completion, subject to meeting the specific criteria for passing the course. The main Coursera pricing model for specialization

	no access is given to summative assessments.	is a subscription model of 39-89US$ (€35-90) per month. Free access to courses and specializations is typically only valid for 7 days and access to course content (no summative assessment) is limited.

Table 2 Common features and guidelines of Coursera and FutureLearn platforms.

Criteria	Common Aspects of MOOCs Platforms
Business models	Sharing of the net revenue of upgrade/subscription fees after operating costs are deducted. Some courses are developed in partnership with, or endorsed by specific companies or organisations. The Harvard CS50x course and data science specializations offered by John Hopkins university on edX and Coursera respectively are notable examples. Business models for courses, specializations/programs may be subscription based, or they may offer a discount for upfront payment if all courses require piecemeal purchase and upgrades.
Specialization/ programme support for linked courses	Coursera and FutureLearn both support the concepts of a series of 4-7 courses that are linked together in order to support advanced and in-depth learning of particular concepts, and application/problem domains. Typically, a final part of a programme/specialization is completion of a large significant summative assessment, often referred to as a capstone project where learners must demonstrate the application of knowledge learned and apply or examine the knowledge in the context of a specific project or idea.
Learning Tools Interoperability Standard	LTI v1.1 is supported by both Coursera and FutureLearn. LTI allows the seamless connection of web-based externally hosted tools and content
Video production advice	Create a course welcome video. HD quality 1080p preferred with high quality audio, 1280x720 minimum resolution. Use a pre-prepared script. Speak directly to learners, using a talking head/above the waist shots of a presenter making frequent eye contact, typically 4-9 minutes length, 20-25 point font slide text to enable easy viewing on mobile platforms, provide subtitles, and a dynamically highlighted transcript of any voice-over. All content should be downloadable. The use of word clouds and concept maps in slides (or on a talking head shot backdrop/blank screen area) can be beneficial to dynamically highlight the current word/concept undergoing explanation. Formative in-video questions/polls can be used to break up a video sequence, and to stimulate interaction, relate and in-video questions/polls to course learning objectives, use multiple slides with large labelled images in preference to one complex image, animate bullet points to appear on slides one at a time, conclude each video with a summary slide.
Staff MOOC content design and production	From our interviews, and reading the literature, it would appear that around 10-18 hours of total staff time is required to design, produce and upload 1 hour of MOOCs content. Additional time delays must be

times	factored in to include any internal and external QA by the organisation associated with the course development and the MOOCs platform providers. Developing a MOOCs course is not a cheap option in terms of staff development time and costs.
Quality Assurance	Both FutureLearn and Coursera have QA guidelines concerning content that will be checked before a course can go live. Courses must work across a wide range of mobile devices, browsers and operating system platforms.
International academic credit relationships	2 ECTS = 4 UK = 1 US/Canadian Credits, 1 year UK Masters is typically 60 ECTS, 120 UK, 30 US/Canadian credits
Approximate costs per academic credit of MOOCs versus traditional Masters	From a brief overview of the stated upgrade costs of FutureLearn courses and organisations awarding FutureLearn linked degrees (Deakin University), we have seen that the approximate costs varies between €110-€200 when considering the cost for a single unit (unrelated to a degree) and an entire Masters programme. The costs at University of Manchester for a 60ECTS credit masters in Computer Science range from €190-€400 dependent on a student's EU/international student status.

8. Guidelines for Implementing a MOOCs Course

This section gives guidelines on how to convert a traditional lecture course into one that is suitable for distance learning and for deployment as an MOOC. It is important to note that university courses of one semester of approximately 10-12 weeks are longer than a typical MOOC course lasting 5-8 weeks. Consequently many institutions create a series of MOOC courses to deliver equivalent learning content to a university course or module. The material in this section summarises content from the following resources, Designing an Online Course, and (MIT News FIXME). The creation of online teaching material should be treated as a project and is generally created using a "5-D", *Define, Design, Develop, Deploy, Deliver* process as outlined in Figure 2 below. Note that although we present guidelines for converting an existing course, it is first important to decide why it is necessary to create MOOC course, and any strategic organisational objectives that need to be met as it will be necessary to consider these objectives as belonging to stakeholders with an interest in the course during the initial *define* process. For example, a short list of strategic organisational objectives might include:

- The need to foster and improve the public understanding of university research outcomes.
- To offer industrially relevant continuing professional development CDP courses.
- To strengthen and improve organisational branding and publicity.
- To improve economics by reducing costs of delivering courses (such as those offering standard support in research methods, and the use of basic programming tools/skills), and increasing fee revenues.

- To improve educational outcomes such as widening participation, student engagement and retention.
- To bring technological innovation to teaching and learning practice.

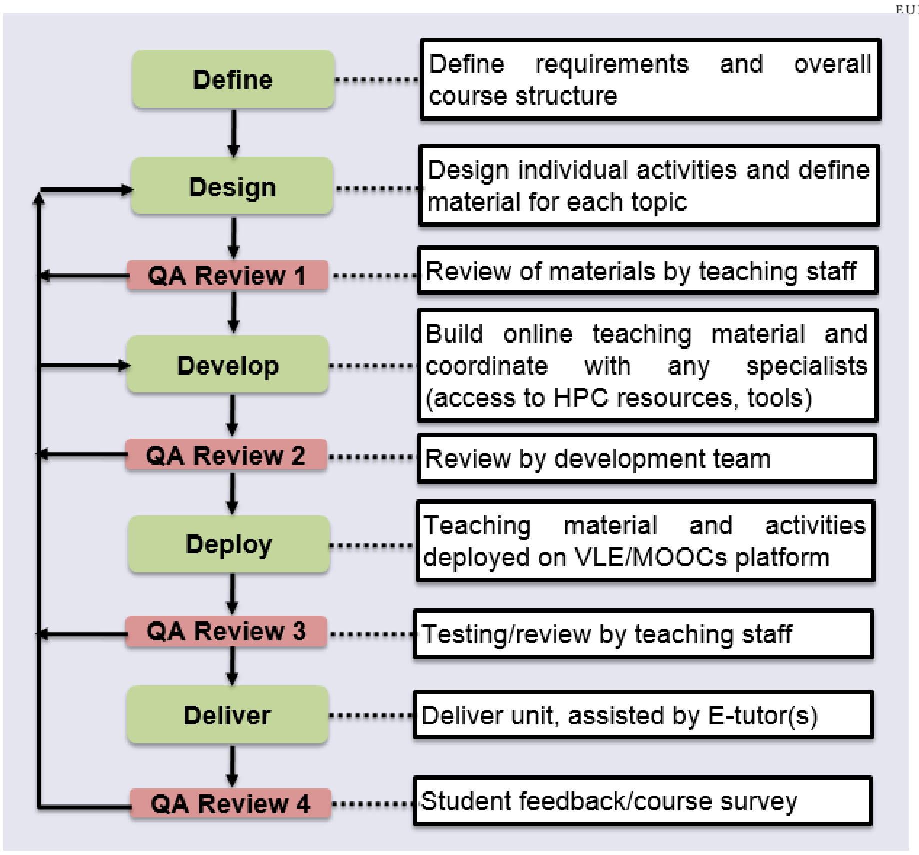
In our investigations, we found that many European institutions created their initial MOOCs courses for two main purposes:

- Cutting-edge "beacon" research subjects that generate a high-level of interest and publicity for the institution.
- Recruitment for established distance learning courses.

In most cases, once the initial courses had been developed, the MOOC material was deployed within the institutions conventional courses in some way, mitigating some of the cost of development.

Define

Course definition refers to the overall requirements and structure of a course with respect to a program of study, such as an MSc, or as a stand-alone aspect of *continuing professional development* (CPD). In the context of converting a single semester course into a distance learning course for a MOOC, it may be desirable to split the unit into two or more linked courses as is common with current MOOCs offerings (The University of Manchester equates 3-4 MOOC courses to a 5 ECTS credit module). Note that some providers do not do this, as they run MOOCs courses in parallel with, and as a direct, or nearly like-for-like distance replacement for their face-to-face courses such as with Harvard's CS50X on edX. Course definition should determine timescales, resource requirements (teaching/technical personnel and hardware/software), and any business model to be followed based on market research as follows.



Define
Define requirements and overall course structure
Design
Design individual activities and define material for each topic
QA Review 1
Review of materials by teaching staff
Develop
Build online teaching material and coordinate with any specialists (access to HPC resources, tools)
QA Review 2
Review by development team
Deploy
Teaching material and activities deployed on VLE/MOOCs platform
QA Review 3
Testing/review by teaching staff
Deliver
Deliver unit, assisted by E-tutor(s)
QA Review 4
Student feedback/course survey

Figure 2: The 5-D Course Development Process modified from (Team, n.d.)

1. **Identify all relevant stakeholders, and their objectives, roles and responsibilities in the course/program.** The stakeholders may range from organizations providing funding support down through to the students that enroll on the course. The *Responsible, Accountable, Consult, Inform* (RACI) model, is a useful tool for identifying roles and responsibilities of stakeholders. A sample set of roles is outlined briefly outlined below. Note, the mapping onto specific personnel in a university, or a technical HPC training/research center will be dependent on local organizational issues. In the context of a Eurolab envisaged pan-European HPC training distance learning provision. It is likely that it will be necessary to define project managers at each organization participating in the project.
    - Course Leader - ultimately responsible for course delivery.
    - Course Tutors - responsible for specific course elements/topics.
    - Project Manager - overall coordination and communication.
    - Learning Designer/Technologist - assists in the design of distance learning activities including both structure and content.
    - Developer - builds and creates online materials and resources.
    - Course Administrator - primary student contact.
2. **Project management activity at the course/program level.** In general such activities will be the responsibility of the project managers, but they will require information gathering from all parties involved in the project.

- Scoping and requirements definition for input to the design phase. For example in terms of the Bologna ECTS credits that the course is expected to deliver, and the entry level qualification requirements of students.
- Planning and milestone setting of completion timeframes for the course define, design, develop, deploy, deliver and quality assurance phases.
- Risk analysis and mitigation planning. For example to cover illness, equipment and technical failures.

Design

A good design is vital when developing a course, as it defines not only the learning objectives and desired outcomes but also the narrative thread through the course and also social interactions between participants which are vital for implementing the pedagogic techniques described in the previous section. Here, we define a six step guide to course design:

Step 1: Collate all current and previous course teaching materials

Materials such as lecture notes, videos, screencasts, reading lists, lab exercises, and details of all existing formative and summative assessments should be collated. It is important to note that designing a distance/e-learning course takes a significant amount of time and 1.5 working days are estimated to be needed for the design, production, and upload of material equivalent to a 1-hour lecture, when supporting videos/screencasts of material, and a formative assessment such as a short online quiz/multiple choice question activity are required. Therefore, approximately 10-12person hours are required for 1 hour of content and assessment. Developing an entirely new course from scratch would take significantly longer. If automated programming assessment/evaluation tools and techniques are to be used, then this will also add to course development time. The time taken to convert a traditional one semester 12 lecture hour module into a distance learning module is therefore in the region of 18-20 working days of effort.

Step 2: Define the main *outcomes* of your course by identifying the main learning objectives and transferable skills to be assessed and developed

Typically, no more than 3-5 learning objectives should be specified for a single course. A *learning objective* is a measurable outcome statement that captures the specific skills, knowledge, and attitudes that learners should be able to exhibit following course completion. Each course outcome is then *mapped* and related to the collection of concepts or *topics* to be studied during a course. Transferable skills, and/or demonstrable competencies are sometimes defined as course outcomes, in addition to learning objectives, in the context of "after completing this course you will be able to perform XYZ". Topics are derived from examining textbooks and reviewing the collated course materials. Note that a single topic or concept will not always have a one-to-one mapping with a specific learning objective/transferable skill. A suggested total number of topics is a maximum of around 5-7 in order to ensure that the topics are suitably focused without becoming too narrow, or too wide in scope. *CompendiumLD*, *Popplet*, and *Mindmeister*, are three tools that can be used to graphically express and capture relationships between concepts/topics, learning objectives and transferable skills in order to produce a "mind or concept map" for later use in course design and delivery. Tools for the automatic production of concept maps, such as leximancer can be used to analyze text based documents using content analysis, and then to produce an image that displays an overview of the extracted semantic information as a concept map.

Step 3: Define a Course Overview

The content in each topic should now be expanded, in order to broadly define a course overview that typically includes:

- Learning objectives and transferable skills.
- Syllabus (content description) and links to the Materials/Technology (textbook chapters, article readings, lectures, digital tools, videos, podcasts, etc.)

Step 4: Define Social Interaction (formative/summative assessments/assignments, e-discussions, online chat tutorials, etc.)

As discussed in the previous section, encouraging social interaction between participants increases student comprehension and completion rates. The potential set of interaction that can take place may be limited by the VLE, and any external tools that are available. In the context of HPC related courses we expect that a combination of web based tools, virtual machines, and traditional networked access and queuing systems will be used to manage remote access to HPC resources. Activities such as Polls with immediate public results (where learners are sent to a short survey (often one question) with a multiple choice question) help learners to gain insight into the views of their peers, and can be used to stimulate discussion in forums and discussion boards. Collaborative and interactive activities can include the *generation of word clouds/concept maps* where learner's most commonly submitted words/phrases are displayed in the largest font. *Interactive pinboards* generalise the word cloud/concept map to include the submission of text and images from a predefined content set. Interactive simulations – perhaps sourced from reusable multimedia content can enable learners to manipulate objects and change variables. Such simulations have been widely used to illustrate the principles of operation of cache systems in computer architecture courses. Information about activities to foster social interaction can be found in Gilly Salmon's web pages.

Step 5: Align the course materials and all course activities to learning objectives and transferable skills

In order to maintain the narrative throughout the course, it is vital that all the course materials are *aligned* to learning objectives. Any activities that are not directly related to the course outcomes will inhibit the narrative flow throughout the course and may reduce student comprehension. Alignment is expected to be an iterative process, and if it is not possible to relate an aspect of the course materials to a learning objective or a transferable skill then a choice must be made:

- The course aspect should be identified as supplemental material.

- The learning objective/transferable skills and/or the course aspect needs revision or removal.

Annotation of a concept map with alignment information can help to identify strengths and weaknesses in a course's design, and an annotated map also assists students in understanding the relevance of specific content to achieving learning objectives and gaining transferable skills. Clearly having many topics, and many course outcomes makes it difficult to produce a coherent concept map. Annotated concept maps can be time sequenced, and this provides an outline course schedule and alternative learning pathways for students. *CompendiumLD* and other similar tools, facilitate annotation of a concept map with learning outcomes and sequence views of the annotated information that indicate potential learning pathways for students.

Step 6: Establish clear formative and summative assessments

Summative assessments will be required for a completion certificate when student learning is to be measured against the achievement of learning objectives. *Formative assessments* monitor student learning via formal and informal processes in order to help students measure their progress, and to guide their learning, and also to inform instructors if a specific topic requires additional clarification or support. For example, if a student performs poorly on a specific formative assessment then they can be guided to review appropriate course materials based on any identified gaps in knowledge. If a specific percentage of enrolled students fail to engage in an assessment, or they fail to achieve a baseline threshold mark then an instructor should take action.

Unfortunately, designing appropriate formative assessments that engage students for distance learning courses is even harder than for the traditional delivery mode of units in large lecture theatres. *Classroom assessment techniques* (CATS) are typically ungraded activities that can be used in a distance learning setting to check that students are engaging with a course and comprehending content. Example assessment styles that can be produced using the IMS Question and Test Interoperability (QTI®) specification are listed here (QTIWorks, 2016). The QTI specification enables the exchange of tests and results between authoring tools, virtual learning platforms, assessment delivery systems and scoring/analytics engines. A table of student assessment and feedback tools and technologies evaluated by UK JISC funded projects are listed here. *VPL- Virtual Programming Lab* is an activity module for the Moodle VLE that manages the submissions and automated assessment of programming assignments. VPL has support for similarity searching and water marking of submitted code in order to detect plagiarism and it has inbuilt support for many programming languages and a secure execution environment. Scripts can be used to test for correctness and to generate assignment marks and student feedback information.

Peer assessment is very valuable, particularly in MOOCs with high numbers of student participation where it may be the only viable way to implement formative assessments. It is also a useful tool to generate social interaction between participants and foster knowledge construction. Notable peer and self-assessment technologies such as PeerWise enable students to both create, and to explain, their understanding of course related assessment questions, and to answer and discuss questions created by their peers.

Questions can be rated and a participation score is generated that can then be used to earn basic, standard and elite badges. PeerWise is similar to peer review discussion boards such as stackoverflow.com, and the professional support communities provided by Intel and ARM Holdings. PeerMark is a *turnitin* peer review assignment tool. Here, instructors can create and manage assignments that allow students to read, review, and evaluate one or many papers submitted by their classmates. Instructors can choose whether the reviews are anonymous or attributed, and can decide if students should be excluded from reviews, or pair students to assign students specific papers to review. Our investigations showed that creating successful peer assessment can be challenging, as students can become disenfranchised if they consider their peers did not engage sufficiently in the assessment process. Successful peer assessment schemes we have studied used very short exercises with clear problem scenario and rubric, thereby reducing the disparity in effort amongst the participants which may be the cause of conflict.

Learning analytics tools enable student engagement, and the probability of student completion to be predicted. *The LAEP Inventory* contains an inventory of learning analytics tools and policies that are in development. For example, Course Signals is a predictive learning analytics system originally produced at Purdue University in the USA. The system uses student data to predict those who are at risk of not successfully completing the course. By using predictive modelling of student data and activity in the learning management system (LMS), each student is assigned a 'risk group,' the colors of which are those of a traffic signal – red, yellow, or green. Further information on learning analytics is available in this ACM article (From MOOCs to Learning Analytics: Scratching the surface of the visual, 2016).

Develop

The develop phase will undertake the construction of the flexible distance learning course materials and any additional information required for technical/administrative and teaching assistant support personnel such as for the administration, on-going maintenance and support of the course and its resources.

1. **Build online content in line with course design and any MOOCs platform defined templates.** The selected MOOCs provider will generally enforce a specific pedagogy, content guidelines, and quality assurance procedures that must be passed if a course is to be allowed to go live on their platform.
2. **Create and test formative activities** across a wide range of devices and operating systems. QA dictates that content delivery should be tested across a range of devices. All activities should be related back to learning objectives, transferable skills and any measurable competencies that are expected to be developed by completing the course.
3. **Create online summative assessments**. If it is not possible to create suitable assessments that test the learning objectives and any measurable levels of competency, then the learning objectives of a course should be reconsidered within a new course design iteration.
4. **Co-ordinate specialist resources and integrate into material, for example tools to remotely access HPC resources.** Approaches such as in-browser, and Docker based containers for the execution of programs and applications in the cloud make it simple for learners to access to traditional computational resources without installing software locally. However, whilst opensource in-browser, and Docker

github examples exist, they still need to be altered to directly interface to the specific HPC resources in question, such as with WebGPU, and/or to interface to a simulator or actual resources.

5. **Identify and test collaboration tools such as message/discussion boards, and polls.** It is important to ensure that the tools work according to any QA procedure standards, and also that suitable support is available to ensure that comments can be moderated in accordance with any learner code of conduct regulations.

Deploy

The deploy stage is heavily related to the technical choices made concerning the MOOCs platform/VLE and any outside tools or technologies which must be integrated. Activities at this stage concern:

1. **Technical - setting up and installing or uploading course materials on FutureLearn, coursera, or a similar MOOCs/VLE.** Normally courses are uploaded and tested on platforms using QA by the course developer, prior to a final QA by the platform provider.
2. **Administrative - registering staff, and providing mechanisms for student registration.** If the course is run as a university course delivering ECTS or equivalent credit, then it will typically be necessary to interface with the university student management systems, perhaps by using an accepted form of external online ID verification such as OAUTH2 and SHIBBOLETH, or those provided by companies such as Jumio. Learners may be required to register both with a university and with an MOOC provider.
3. **Training - including the training of professional/academic teaching staff, and any administrative, teaching assistants or tutors that might be deployed that must use and access the chosen MOOC platform.**

Deliver

Course delivery is the final step. Ideally, all course materials will be hosted online and tested prior to delivery. The key teaching activities for course tutors and teaching assistants will be

1. **Driving online collaboration using planned discussions released at key points to stimulate interaction targeting specific course content and aspects of assessment.** It is important to ensure that any comments and discussions are moderated, and any queries from learners, and that any course issues or errors related to course delivery are dealt with as promptly as possible.
2. **Providing feedback, and monitoring any learning analytics information that may be available concerning student feedback and engagement.** A common feature of MOOCs courses is a relatively low completion rate for courses that are completely free; consequently it may be necessary to screen out non-engaging students when performing statistical analysis on engagement and on the achievement of learning objectives by the student cohort. The low engagement rates will also have to be considered and managed when designing and delivering any online collaborative activities and assessments.

Quality Assurance

Quality assurance reviews should examine the deliverables and documents produced after each phase in the process in order to ensure the project is proceeding with suitable high quality that meets its definition and specification. The project manager(s) should be responsible and take leadership of this activity

1. **QA Review 1 (after Design Stage).** A full outline of the course structure and content should be available, and it should be possible to define a firm project plan including consolidation of any milestones.
2. **QA Review 2 (after Development Stage).** The full set of course resources should now be available for checking by internal peer review by the development team and any external assessors. The goal is to ensure consistency of quality, and that any alignment with ECTS credits is correct.
3. **QA Review 3 (after Deployment Stage).** The full course is available for review on an VLE platform. It is useful to conduct user acceptance testing with a focus group comprised of sample students/learners and teaching professionals in order to gather their comments and opinions.
4. **Final Review (after/during delivery).** This is intended to deliver a continuous process of improvement for the course by gathering feedback from course participants and teaching staff in order to refine and enhance the online material and content. The ongoing support and maintenance costs of this review should be built into the project requirements.

The main MOOC platforms all include quality assurance steps that must be passed before a course can go live. The exact nature of the QA steps is dependent on the platform, but delays of around 30 days are expected for QA tests by beta testers and MOOCs partnership managers. It is important to factor in these delays, and the time taken to fix and re-evaluate any QA issues arising that might prevent an entire course, or even an important aspect of summative assessment from going live in a course.

Practical Tips for Creating an Online/Distance Learning Course Relating to HPC

MOOCs providers, and/or your organisation may insist on the usage of specific course templates, look-and-feel interfaces, and one may be limited to specific type of assessments. It may also be necessary to comply with any quality assurance, and/or pedagogic styles – such as narrative storytelling for course design. The practical tips outlined below were mainly gained by examining the course building information and assets provided for partners of Coursera and FutureLearn MOOCs providers, and by examining

1. **Look at sample courses in your own organization, and courses with similar or related content:**- Merlot, edEx Courses, The MIT Open Courseware Initiative, Coursera, HPC University, Udacity, FutureLearn, and Open2Study.
2. **Identify and develop high quality aligned content materials that directly matches the outcomes of learning objectives and transferable skills:** such as the OER Commons, Community College Consortium for Open Educational Resources.
3. **Develop accessible materials that are designed for universal delivery.** Course content needs to be accessible and tested on mobile devices, screen readers, laptops and desktops, as well as being, able to meet the needs of all learners, including those with disabilities. Universal Design best practices are covered in these links: MCC Center for Teaching & Learning, Creating Accessible Course Materials, University of Northern Colorado's Universal Design for Learning Tutorial , and the VideoHelpForum which contains practical information on the production and the streaming of audio/video multimedia content.

- Screen reader support requirements for documents containing text and diagrams. Text documents will appear to screen readers as a long list of unstructured text unless appropriate formatting is applied to indicate headings. Diagrams and photos require alternative text to be provided to describe the information contained in an image to screenreader software.
- Video should be shot in landscape mode, at HD resolution of 1920x1080 pixels or better. Simple talking head shots above the waist are preferable, with durations limited to 7 minutes or less where possible in order to enable learners/students to dip in and out of course content in multiple short sittings without losing interest. Video/image scenes should be clean, clear of clutter and free from background noise. It will often be necessary to use an external microphone in order to record high quality audio. The copyright of any images/footage used should be checked, and it is generally necessary to seek permission to shoot on location. It is good practice to get any presenters to sign an image release form before any content is recorded.
- Screencasting/recording with an audio voiceover can be achieved using software such as Camtasia, Adobe Presenter, Quicktime, recordMyDesktop, and ffmpeg.
- Always use a pre-prepared script for recording audio and adding any commentary to screencasts. Transcripts and/or subtitling should be made available in order to ensure material remains accessible.
- Absolute dates such as 30th June 2017 should be used in preference to 30th June this year, in order to avoid unnecessary updating of materials.
- Test your content works correctly across the full range of desktops, tablets, laptops and mobile devices that are supported by your MOOCs/distance learning provider.

4. **Use automated assessment and assignment techniques**: the assessments and assignments should give structured feedback, rather than just a grade. Unit testing, coding standard metrics, and randomised data inputs for programming based assignments are just some aspects of automated testing that could be deployed when scaling to large MOOCs courses with hundreds and thousands of students. Summative assessments need to be related to course learning objectives and to any measurable performance and competencies or skills that are relevant.
    - Assessments within MOOCs typically include multiple choice quizzes and tools/techniques to ensure that learners/students have both engaged, and understood the material presented in course units, steps and modules. For quizzes, each answer choice should provide automatic elaborative feedback to explain why an answer is correct or incorrect. For summative quizzes, the elaborative feedback must be tailored to avoid revealing the correct answer. The feedback should explain how to approach solving a problem, and also where to find more information about the topic in the course. Such elaborative feedback helps learners to better understand content, and to grasp new knowledge.
    - Active learning requires engagement, this may take the form of leaving a comment or a response to a question, and the ability to participate in moderated discussion boards attached to course content. On the FutureLearn platform each step in a course needs to be marked as complete, and learners/students can easily check their progress on a unit. Coursera

recommends the use of discussion prompts, polls and in-video questions to provide punctuation points for active engagement by a learner. The goals are to help learners to assess their own progress and to help connect new course content with any prior content. Coursera and FutureLearn support for programming assessments are described in the relevant headings in the previous section.

Non-automatic summative assessments can either be issued using peer-review, and/or requiring marking by teaching assistants. Such assessments are normally only available to learners that have paid to obtain certificates of achievement/assessment. The goal of peer review assessments is to enable learners to develop evaluative judgement and to apply new knowledge to creative open-ended problems beyond the current capabilities of automatic assessment tools. It is important to be able to moderate any free text comments that can be entered by learners undertaking a review. Non-automatic summative assessment can also be supported by linking to external web sites and tools using the LTI standard, and/or using secure web links to services such as turnitin for essays. Exams can be catered for using in-person test centres, or online with ID verification. List of freely available LTI tools are available on websites such as the one eduappcenter, imsglobal, and a github resource set of PHP classes for LTI tools provides an initial starting point if it is necessary to create a bespoke LTI tool interface to specific hardware or software.

5. **Blended learning integration of distance learning and face-to-face interactions**: for non MOOC courses with smaller student cohorts (such as CPD courses offered by university research centres and HPC organisations). It may be useful to schedule short intensive face-to-face events (typically lasting 1-2 days) at periodic intervals in conjunction with distance learning delivery. Video-conferencing tools and technologies may be used to deliver, record and edit face-to-face sessions within a distance learning setting. Webinars, consisting of a short presentation, followed by an interactive discussion are a great way to encourage interaction between participants as well as instructor/student interaction. Such delivery modes have been used on professional doctorate courses and are commonplace for summer schools and workshops for building communities around specific aspects of research and technology.

Sample Courses & Resources Matching the HPC Curriculum

1. Heterogeneous Parallel Programming is a University of Illinois course dealing with massively parallel programming mainly using CUDA and later with OpenACC, C++AMP and OpenCL using an online web-based resource www.webgpu.com that enables development and execution of GPU programs in the cloud, and additionally for local development to take place using a library. NVidia, AMD and Khronos provide extensive teaching resources available from the following CUDA and AMD OpenCL, Khronos OpenCL links.
2. High Performance Scientific Computing is a University of Washington course that provides an introduction to efficient serial and parallel computing using Fortran 90, OpenMP, MPI, and Python, and software development tools such as version control, Makefiles, and debugging. The course allows students to undertake quizes that are automatically graded and homework assignments (that are not graded) using either a prepared virtual machine image, or Amazon Web Services EC2 AMI.

3. MIT Computer Architecture 2005 Graduate Course, OERCOMMONS course.
4. HPC teaching material contain lists of OERCOMMONS resources matching the HPC search term
5. Multithreading teaching material OERCOMMONS course on multithreading and multiprocessing.
6. OERCOMMONS search link

9. Relationship with Existing and Proposed Curricula

In this section we compare our approach to similar approaches in the community. This is first of all a sanity check, but also highlights similarities and differences in the approach and emphasis of our curriculum. We discuss curricula and HPC educational initiatives by PRACE, HRLS, EIT ICT Labs and ACM. We do not discuss the Intel-proposed curriculum for parallel systems, as it is geared towards practical use of Intel specific platforms. Also, the ETP4HPC Strategic Agenda does not explicitly address educational matters or activities, but refers to EXDCI and Eurolab4HPC for such activities. However, EXDCI is not yet involved in educational activities.

Relationship with ACM Computer Science Curricula 2013

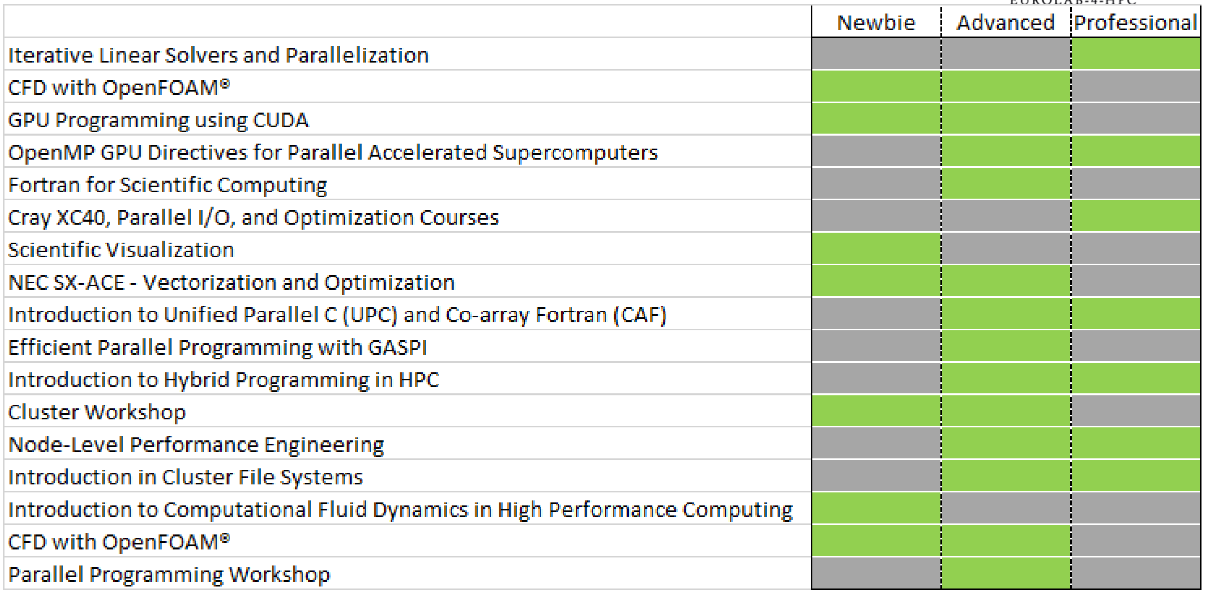
The ACM Computer Science Curricula 2013 (http://www.acm.org/education/CS2013-final-report.pdf) offers curriculum guidelines for undergraduate degree programs in Computer Science. Being focused on a traditional 4-year program, its advanced topics overlap with MSc level in Bologna-aligned European programs. The latest curricula proposal vastly upgraded the coverage of parallel thinking proposing topics such as:

- Parallel and Distributed Computing
- Parallelism Fundamentals
- Parallel Decomposition
- Parallel Algorithms, Analysis, and Programming
- Parallel Architecture
- Parallel Performance
- Distributed Systems
- Cloud Computing

In the previous version (2005) only Distributed Systems were proposed. These topics are very much in line with the courses listed above. Since this is a proposal for undergraduate degrees, the level of specialization is understandably lower, and the suggestions concentrate on core technologies.

Course Offers by HLRS

In the scope of existing HPC Curricula HLRS offers a set of various courses to industry and academia. The HLRS courses are about various topics and technologies needed for simulation purposes. Most of them are performed every year. However, the HLRS course program is flexible in a way that courses are evaluated very carefully and updates on existing courses as well as the introduction of new courses are possible.
The following table gives an overview of the topic range and levels of HLRS courses (Figure 3).



	Newbie	Advanced	Professional
Iterative Linear Solvers and Parallelization	▨	▨	🟩
CFD with OpenFOAM®	🟩	🟩	▨
GPU Programming using CUDA	🟩	🟩	▨
OpenMP GPU Directives for Parallel Accelerated Supercomputers	▨	🟩	🟩
Fortran for Scientific Computing	▨	🟩	▨
Cray XC40, Parallel I/O, and Optimization Courses	▨	▨	🟩
Scientific Visualization	🟩	▨	▨
NEC SX-ACE - Vectorization and Optimization	🟩	🟩	▨
Introduction to Unified Parallel C (UPC) and Co-array Fortran (CAF)	▨	🟩	🟩
Efficient Parallel Programming with GASPI	▨	🟩	▨
Introduction to Hybrid Programming in HPC	▨	🟩	🟩
Cluster Workshop	🟩	🟩	▨
Node-Level Performance Engineering	▨	🟩	🟩
Introduction in Cluster File Systems	▨	🟩	🟩
Introduction to Computational Fluid Dynamics in High Performance Computing	🟩	▨	▨
CFD with OpenFOAM®	🟩	🟩	▨
Parallel Programming Workshop	▨	🟩	▨

Figure 3: HLRS Courses and their level

As presented, course levels are defined into three categories for enabling attendees to select a best fitting course considering the individual technology skills. However, the course levels are recommendations and decision guidance. A description of the HLRS courses is given in the following by giving a short description of each course.

Iterative Linear Solvers and Parallelization
The focus is on iterative and parallel solvers, the parallel programming models MPI and OpenMP, and the parallel middleware PETSc.

CFD with OpenFOAM®
The five-day workshop gives an introduction to OpenFOAM® applied on CFD phenomena.

GPU Programming using CUDA
The course provides an introduction to the programming language CUDA that is used to write fast numeric algorithms for NVIDIA graphics processors (GPUs)

OpenMP GPU Directives for Parallel Accelerated Supercomputers
This workshop will cover the directive-based programming model based on OpenMP v4 and OpenACC v2 whose multi-vendor support allows users to portably develop applications for parallel accelerated supercomputers.

Fortran for Scientific Computing
This course is dedicated for scientists and students to learn (sequential) programming with Fortran of scientific applications. The course teaches newest Fortran standards. Hands-on sessions will allow users to immediately test and understand the language constructs. This workshop provides scientific training in Computational Science, and in addition, the scientific exchange of the participants among themselves.

Cray XC40, Parallel I/O, and Optimization Courses
In order to help users running efficiently on the new large Cray XC40/Hazelhen system at HLRS, HLRS and Cray offer an optimization workshop for applications running at scale. The intent of this workshop is to tune the performance of such codes by detecting, locating and solving bottlenecks.

Scientific Visualization
This two-day course is targeted at researchers with basic knowledge in numerical simulation, who would like to learn how to visualize their simulation results on the desktop but also in Augmented Reality and Virtual Environments.
NEC SX-ACE - Vectorization and Optimization
The participants learn about the configuration of the NEC SX-ACE system at HLRS and how to use this cluster of vectorizing shared memory nodes.
Introduction to Unified Parallel C (UPC) and Co-array Fortran (CAF)
Partitioned Global Address Space (PGAS) is a new model for parallel programming. This course gives an introduction to this novel approach of expressing parallelism.
Efficient Parallel Programming with GASPI
In this tutorial we present an asynchronous dataflow programming model for Partitioned Global Address Spaces (PGAS) as an alternative to the programming model of MPI.
Introduction to Hybrid Programming in HPC
This course analyses the strengths and weaknesses of several parallel programming models on clusters of SMP nodes, presents tools for hybrid programming such as thread/process placement support and performance analysis and gives attendees the opportunity to try the new MPI shared memory interface and explore some pitfalls of hybrid MPI+OpenMP programming.
Cluster Workshop
HLRS offers a vendor-independent workshop. Topics span from the design of compute clusters to details on different hardware components, operating systems, file systems and modes of operation, touching on specific software solutions. Further, typical problems and strategies for their solution will be discussed.
Node-Level Performance Engineering
This course teaches performance engineering approaches on the compute node level.
Introduction in Cluster File Systems
This course is a one-day workshop for giving an overview and present first steps regarding the architecture, functionality and important varieties of the three cluster filesystems, BeeGFS, IBM Spectrum Scale and Intel Enterprise Edition for Lustre.
Introduction to Computational Fluid Dynamics in High Performance Computing
The course deals with current numerical methods for Computational Fluid Dynamics in the context of high performance computing.
CFD with OpenFOAM®
The five-day workshop gives an introduction to OpenFOAM® applied on CFD phenomena.
Parallel Programming Workshop
Distributed and shared memory parallelization with MPI and OpenMP.
The HLRS program is offered to the community including industrial and academic users. The course offer is available every year. However, changes are possible based on the evaluation process of the courses.

Course Offers by PRACE

The overall goal of PRACE (Partnership of Advanced Computing in Europe) is to support and enable scientific discovery, engineering research and development across multiple disciplines for strengthen the European competitiveness. PRACE consists of a consortium of 25 members representing European Union member states and associated countries. In this scope PRACE

offers courses relevant for the HPC domain in Europe. In general PRACE distinguishes between events, seasonal schools, workshops and private events:
Events: This category includes training events being courses offered by the PRACE consortium
Seasonal Schools: The PRACE seasonal schools are multi-day sessions such as a spring or autumn school
Workshops: Besides the events and seasonal schools, PRACE offers workshops being training sessions for groups of participants for practical exercises
Private Events: This category is accessible for registered users and offers further training sessions
At time of writing of this document the overall amount of PRACE sessions is about 373 training units. Those courses covering following topics:

- Parallel Programming Workshop (MPI, OpenMP and advanced topics)
- Parallel Programming Workshop (Train the Trainer)
- Advanced MPI
- Introduction to High Performance Computing with Fortran
- Introduction to High Performance Computing with C
- Advanced OpenMP
- Message-Passing Programming with MPI
- Hands-on Introduction to HPC
- 7th Programming and Tuning Massively Parallel Systems summer school (PUMPS)
- Introduction to OpenACC
- Performance Analysis Workshop
- Introduction to CUDA Programming

Generally, the PRACE training distinguishes between type of training session (as shown above) and the topic. Further, the provided training units are offered by different partners, thus each of them brings in its expert knowledge. For managing the training session PRACE maintains a web site giving an overview of available training (PRACE, 2016).

Course Offers by EIT

The EIT Digital Master School is a joint initiative by the technical universities and business schools in Europe. It offers two related degrees, Data Science, and Cloud Computing. Since this is a joint effort among several universities, the programs have considerable difference in emphasis and structure. The most related courses are:
Cloud Computing and Big Data Ecosystems Design
Big Data
Technologies for big data
Data mining
Statistics for Big Data

10. Conclusions

This document presented the results of our investigations into the current status and the future of HPC Education and Training. It is our view that, as computing systems around us become increasingly parallel, it is imperative that parallel thinking is introduced to as many students and practitioners as possible. The aim of HPC education is to produce graduates that are skilled and trained in parallel and HPC platforms, programming models, tools, etc. and can tackle and adapt to the challenges of future HPC systems that are likely to be heterogeneous many-core systems. To this end, we developed a curriculum framework to meet the needs of new programmes with specific technological objectives and transferable skills that industry may require in the future. The curriculum was developed after undertaking a survey of existing HPC courses throughout EU higher-education institutes and HPC research institutes. The curriculum was published in M12 of the EULab-4-HPC project.

In order to evaluate the suitability and reach of the curriculum, we undertook a further survey of HPC practitioners to ascertain their feelings about the proposed courses. As a result of their feedback we added a "*Data Science Fundamentals*" course to address the data-science processing explosion. We refocused some targeted courses, on Hadoop and OpenCL, to cover a range of programming frameworks and accelerator architectures. The revised curriculum was presented in sections 2-5 of this document.

The second key part of our investigation was to distil a set of best-practices for developing HPC distance-learning and MOOCs courses in order to facilitate the development of new courses in cutting edge HPC techniques. This was largely achieved through a series of interviews with learning technologists and developers of MOOC course content and course development information from FutureLearn, and Coursera MOOCs platforms. The key findings of our investigation are as follows:

- xMOOCs are the most suitable course type for the deployment of a multi institution HPC curriculum.
- An individual MOOC course is approximately equivalent to 1-1.5 ECTS credits
- Creating a good MOOCs course requires a ranged of skilled professionals (in addition to the subject experts):
  - Educational professionals/learning technologists
  - Video Editors
  - Course Designers
- Course design is a very important activity that should start with the definition of measurable learning outcomes, and objectives concerning the capabilities of learners after successful completion of the course.
- High levels of social Interaction between learners (course participants) reinforces active engagement, and can help to increase MOOC completion rates that are traditionally very low in comparison to standard university degree and *continuing professional development (CPD)* courses.
- Assessments must be carefully designed to encourage social interaction and not distract from the course narrative.
- The development of a good course is an iterative process that requires monitoring after it has been deployed and updated in response to student feedback and any delivery

problems that may arise.

Clearly, there are a number of financial and technical barriers that might prevent or hinder an organisation that seeks to develop and provide HPC related training and development as MOOCs courses. In the action list below, we define forward looking actions concerning potential future work to foster and encourage the development of MOOCs in HPC areas:

- The development of software tool ecosystem support for managing shared virtualised access to specialised HPC and embedded system development platform resources and simulators/emulators would reduce implementation risk for HPC MOOCs. Exemplar open source tools / libraries (for example Docker images) would enable course developers to give learners access to specialised resources with minimal effort.
- To develop a MOOC template and associated practical guidelines, for the production of showcase/exemplar MOOC that demonstrate how to apply best practices, and how to physically realise and deliver an HPC course on MOOC platforms.
- To foster and encourage the development and production of HPC training using MOOCs, via a MOOC prototyping fund to subsidise the financial startup costs of developing and deploying MOOC HPC courses (currently the costs for development and a one-year support period are significant at approximately € 33k). Industry and educational organisations would need to competitively apply to the fund. The intended beneficiaries of the fund are European companies/organisations and end-users with unmet training and recruitment needs.

References

(n.d.). Retrieved June 20, 2016, from http://compendiumld.open.ac.uk/

(n.d.). Retrieved June 20, 2016, from Popplet Web Site: http://popplet.com/

Ally, M. (2004). Foundations of educational theory for online learning. *Theory and practice of online learning*, 15-44.

Anderson, T., & Dron, J. (2010). Three generations of distance education pedagogy. *The International Review of Research in Open and Distributed Learning, 12(3)*, .80-97.

Chauhan, A. (2014). Massive Open Online Courses (MOOCS): Emerging Trends in Assesment and Accreditation. *Digital Education Review*, 7-17.

Course Signals. (2016, June 20). Retrieved from Purdue University: http://www.itap.purdue.edu/studio/signals/

D.F. Ohan, J. S. (2014). Droput Rate of massive open onlnie courses: behavioural patterns. *EDULEARN Proceedings*, (pp. 5825-5834).

Daniel, J. (2012). Making Sense of MOOCs: Musings in a Maze of Myth, Paradox and Possibility. *Journal of Interactive Media in Education*. doi:http://doi.org/10.5334/2012-18

Designing an Online Course. (2016, June 20). Retrieved June 20, 2016, from Centre for Teaching and Learning, Mesa Community College: http://ctl.mesacc.edu/teaching/designing-an-online-course/

Dillenbourg, P., Schneider, D., & Synteta, P. (2002). Virtual learning environments. *3rd Hellenic Conference" Information & Communication Technologies in Education"*, (pp. 3-18).

From MOOCs to Learning Analytics: Scratching the surface of the visual. (2016, June 20). Retrieved from ACM eLearn Magazine: http://elearnmag.acm.org/archive.cfm?aid=2686744

Gagne, R. M. (1970). *The Conditions of Learning.* Thomson Learning.

Garrison, D. R., Anderson, T., & Archer, W. (2001). Critical Thinking, Cognitive Presence, and Computer Conferencing in Distance Education.

Hazel, P. (2008). Toward a Narrative Pedagogy for Interactive Learning Environments. *Interactive Learning Environments, 16*(3), 199-213.

Kanuka, H., & Anderson, T. (1999). Using constructivism in technology-mediated learning: Constructing order out of the chaos in the literature. *Radical Pedagogy, 2*(1).

Koohang, A., Riley, L., Smith, T., & Schreurs, J. (2009). E-Learning and Constructivism: From Theory to Application. *5*(1).

LAEP Inventory - learning analytics tools, policies and practices. (2016, June 20). Retrieved from Cloudworks: http://cloudworks.ac.uk/cloudscape/view/2959

Lane, L. (2008). Toolbox or trap? Course management systems and pedagogy. *Educause Quarterly*, 4.

Mindemeister Web Site. (n.d.). Retrieved June 20, 2016, from https://www.mindmeister.com/

MIT News. (n.d.). Retrieved June 20, 2016, from http://news.mit.edu/2012/mitx-edx-first-course-recap-0716

PeerWise. (2016, Jun 20). Retrieved from Computer Science, University of Auckland: https://peerwise.cs.auckland.ac.nz/

Plowman, L. (1998). Narrative, Linearity and Interactivity: Making Sense of Interactive Multimedia. *British Journal of Educational Technology, 27*(2), 92-105.

PRACE. (2016). *Prace Web Site*. Retrieved July 1, 2016, from https://events.prace-ri.eu/

Project Smart Exploring trends and developments in project management today. (n.d.). Retrieved June 20, 2016, from https://www.projectsmart.co.uk/raci-matrix.php

QTIWorks. (2016, June 20). Retrieved from University of Edinburgh, School of Physica and Astronomy: https://webapps.ph.ed.ac.uk/qtiworks/anonymous/samples#cat1

Salmon, G. (2004). *E-moderating: The key to teaching and learning online.* Psychology Press.

Siemens, G. (2005). Connectivism: A learning theory for the digital age. *Instructional Technology and Distance Education*, 3-10.

Team, E. e. (n.d.). *Distance and Blended Learning Guide v1.1 in the Faculty of Engineering and Physical Sciences , University of Manchester*. Retrieved June 20, 2016, from Teaching and Learning Support Office, University of Manchester, UK: http://www.tlso.manchester.ac.uk/media/services/tlso/content/files/dlwebsite/EPS%20DL%20Development%20Guide%20v1.1.pdf

The Design Studio. (2016, June 20). Retrieved from Jisc: http://jiscdesignstudio.pbworks.com/w/page/67875897/Assessment%20and%20feedback%20technologies%20table

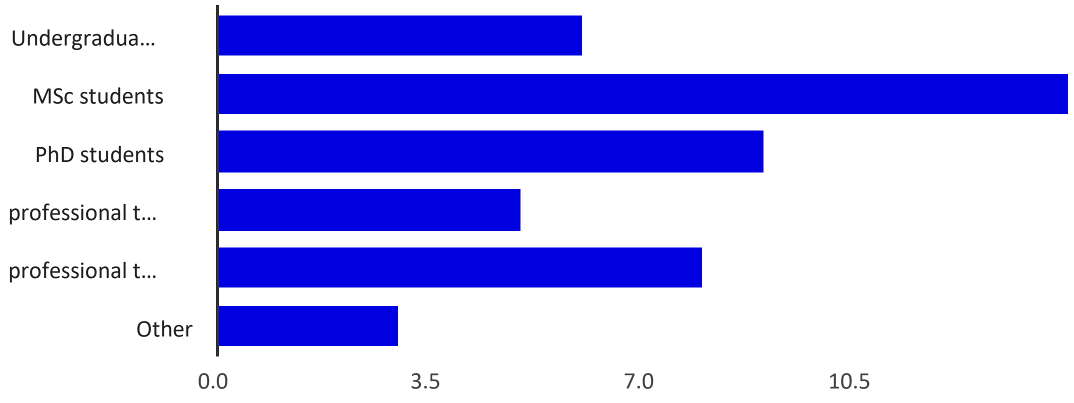
University Teaching & Learning Center. (2016, June 20). Retrieved from Geirgre Washington University:) https://tlc.provost.gwu.edu/classroom-assessment-techniques

VPL. (2016, June 20). Retrieved from The Virtual Programming Lab for Moodle: http://vpl.dis.ulpgc.es/

Appendix: Questionnaire Results

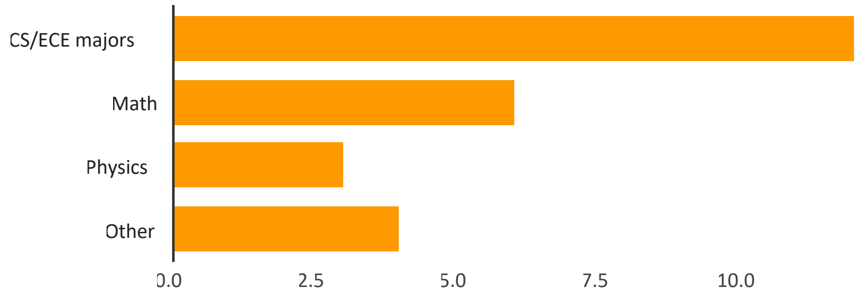
Part A: Curricula Emphasis

1. Target audience:



Undergraduate students	6	35.3%
MSc students	14	82.4%
PhD students	9	52.9%
professional training for BSc/MSc-level expertise in the course subject	5	29.4%
professional training for PhD-level expertise in the course subject	8	47.1%
Other	3	17.6%

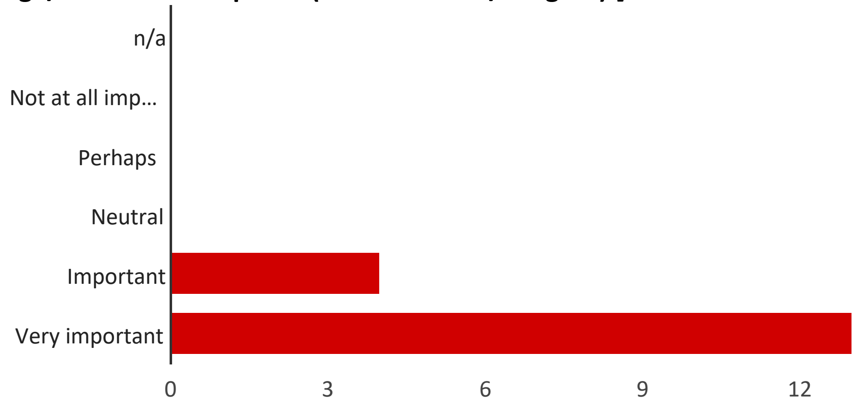
2. Targeted audience expected background



CS/ECE majors	12	70.6%
Math	6	35.3%
Physics	3	17.6%

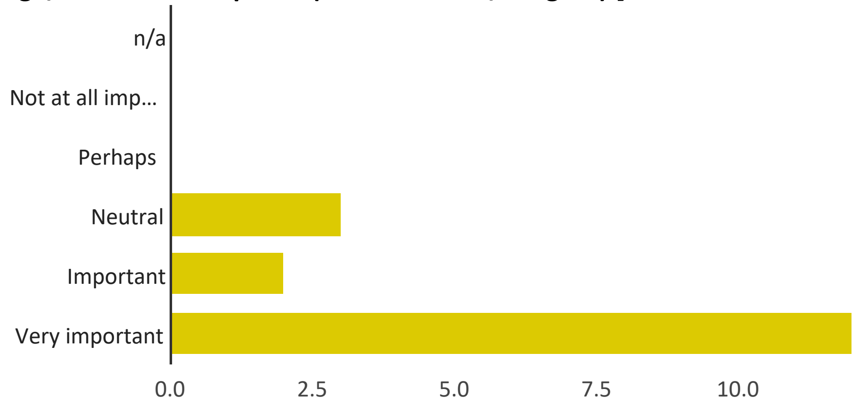
Other	4	23.5%

OpemMP, CUDA, OpenCL, shared memory, Run-time systems, multithreading, multi-core [3. Knowledge/Skills to be acquired (multichecklist/weight?):]



n/a	0	0%
Not at all important	0	0%
Perhaps	0	0%
Neutral	0	0%
Important	4	23.5%
Very important	13	76.5%

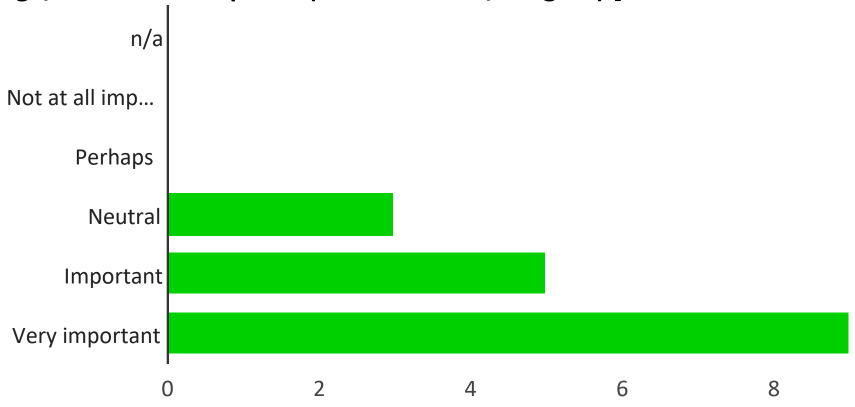
Parallel Algorithms
[3. Knowledge/Skills to be acquired (multichecklist/weight?):]



n/a	0	0%
Not at all important	0	0%
Perhaps	0	0%
Neutral	3	17.6%

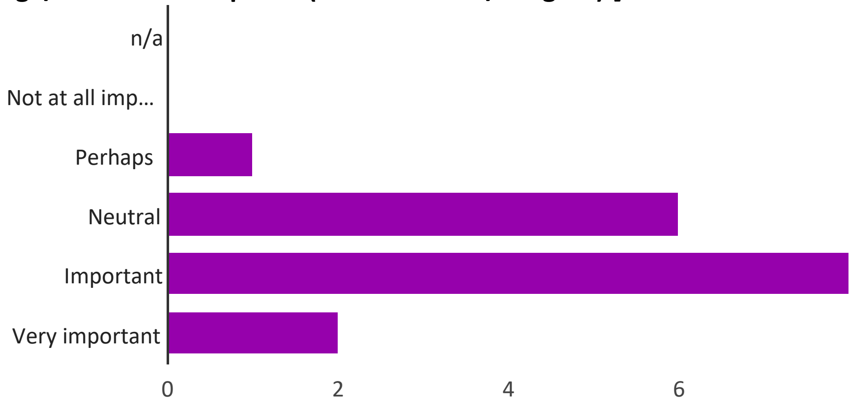
Important	2	11.8%
Very important	12	70.6%

Parallel Data Structures
[3. Knowledge/Skills to be acquired (multichecklist/weight?):]



n/a	0	0%
Not at all important	0	0%
Perhaps	0	0%
Neutral	3	17.6%
Important	5	29.4%
Very important	9	52.9%

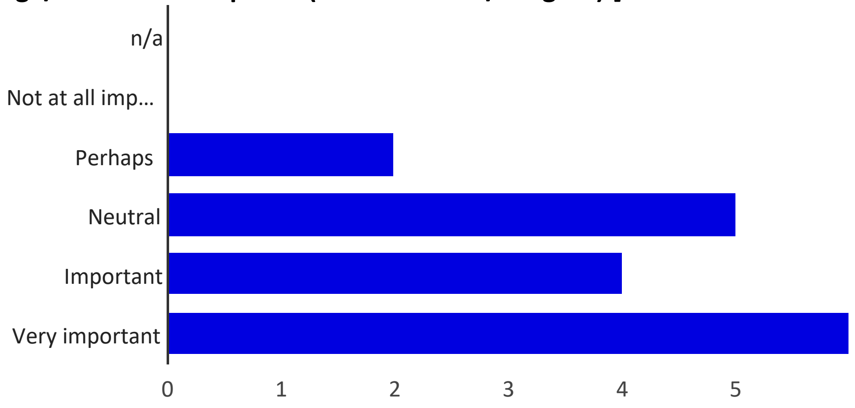
Fault-tolerance
[3. Knowledge/Skills to be acquired (multichecklist/weight?):]



n/a	0	0%
Not at all important	0	0%
Perhaps	1	5.9%

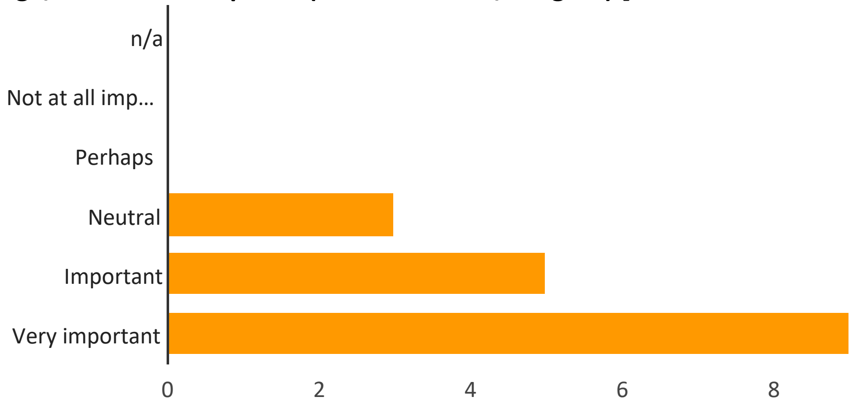
Neutral	6	35.3%
Important	8	47.1%
Very important	2	11.8%

Memory & Consistency models
[3. Knowledge/Skills to be acquired (multichecklist/weight?):]



n/a	0	0%
Not at all important	0	0%
Perhaps	2	11.8%
Neutral	5	29.4%
Important	4	23.5%
Very important	6	35.3%

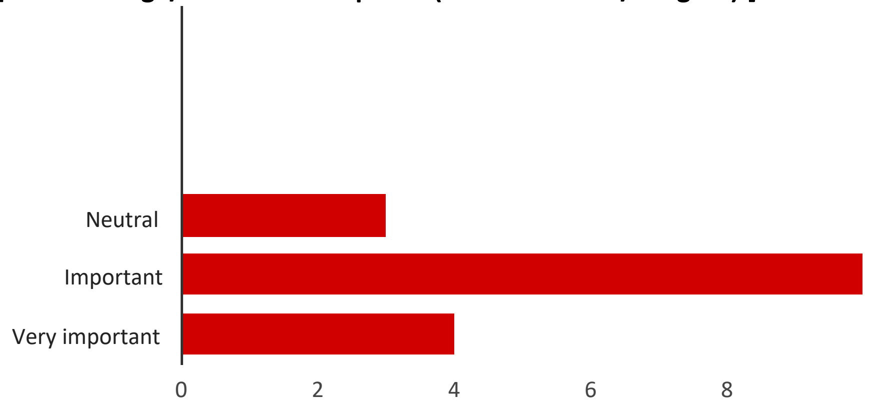
Communication & Synchronization
[3. Knowledge/Skills to be acquired (multi-checklist/weight?):]



n/a	0	0%
Not at all important	0	0%

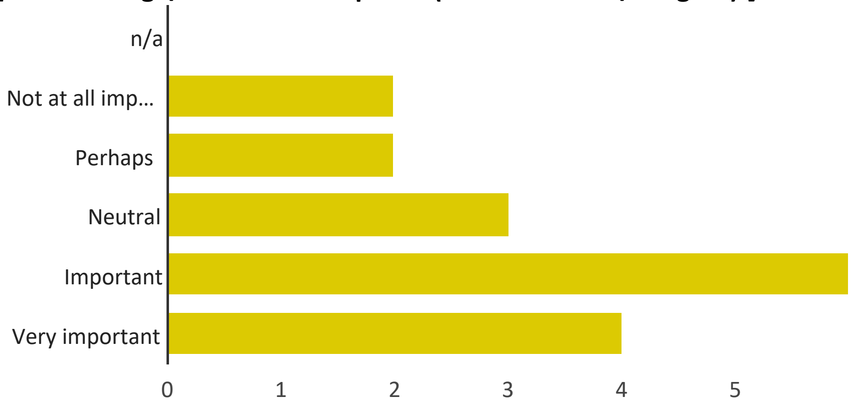
Perhaps	0	0%
Neutral	3	17.6%
Important	5	29.4%
Very important	9	52.9%

Performance Analysis and Tuning
[3. Knowledge/Skills to be acquired (multi-checklist/weight?):]



n/a	0	0%
Not at all important	0	0%
Perhaps	0	0%
Neutral	3	17.6%
Important	10	58.8%
Very important	4	23.5%

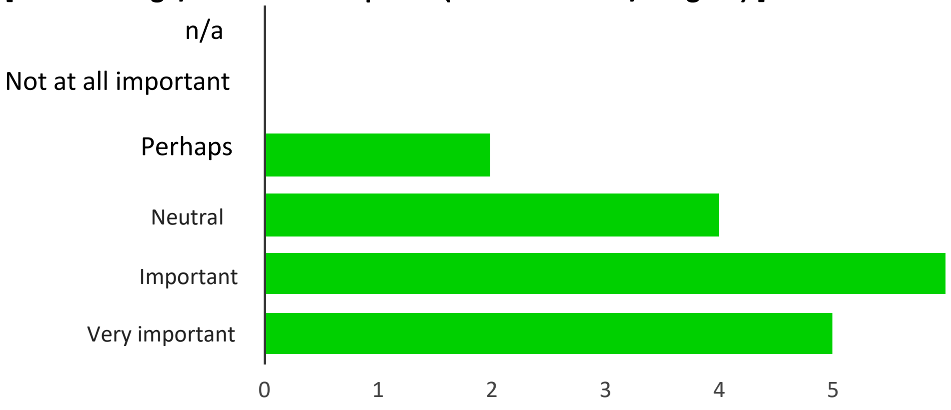
Parallel I/O
[3. Knowledge/Skills to be acquired (multi-checklist/weight?):]



n/a	0	0%

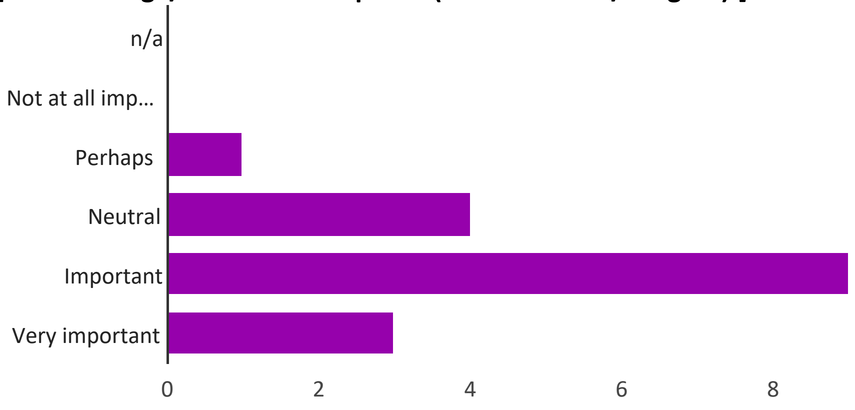
Not at all important	2	11.8%
Perhaps	2	11.8%
Neutral	3	17.6%
Important	6	35.3%
Very important	4	23.5%

Debugging
[3. Knowledge/Skills to be acquired (multi-checklist/weight?):]



n/a	0	0%
Not at all important	0	0%
Perhaps	1	5.9%
Neutral	4	23.5%
Important	6	35.3%
Very important	5	29.4%

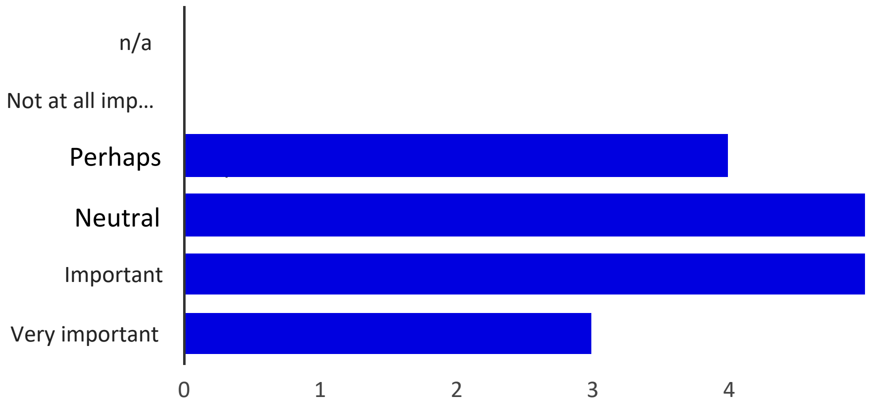
Performance debugging
[3. Knowledge/Skills to be acquired (multichecklist/weight?):]



n/a	0	0%

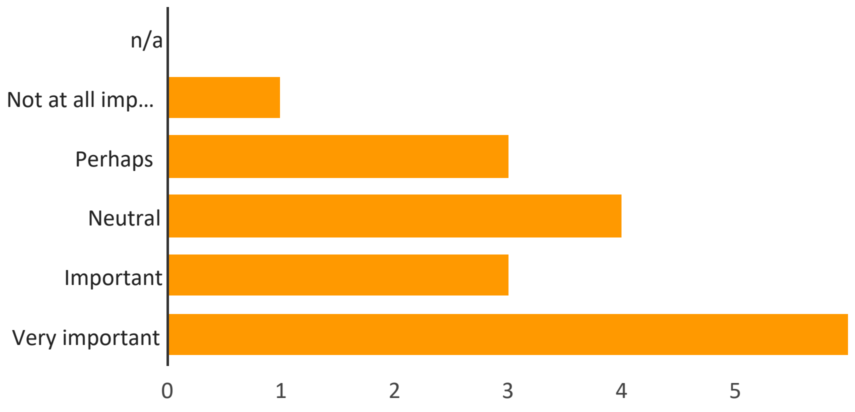
Not at all important	0	0%
Perhaps	1	5.9%
Neutral	4	23.5%
Important	9	52.9%
Very important	3	17.6%

Visualization
[3. Knowledge/Skills to be acquired (multi-checklist/weight?):]



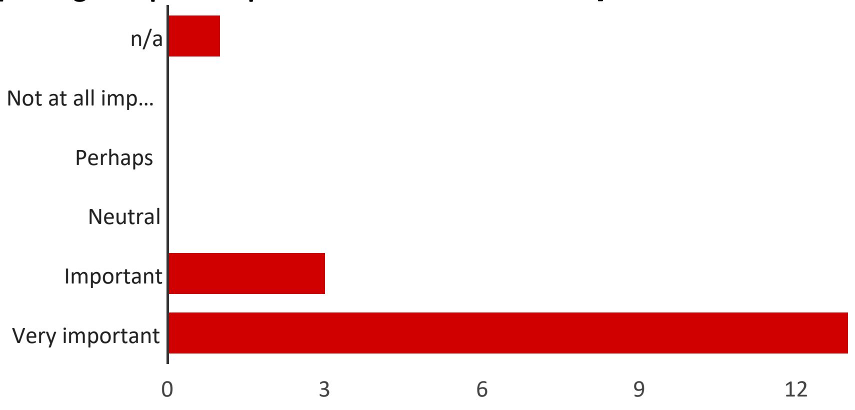
n/a	0	0%
Not at all important	0	0%
Perhaps	4	32.5%
Neutral	5	29.4%
Important	5	29.4%
Very important	3	17.6%

Compilation/Compiler technology
[3. Knowledge/Skills to be acquired (multi-checklist/weight?):]



n/a	0	0%
Not at all important	1	5.9%
Perhaps	3	17.6%
Neutral	4	23.5%
Important	3	17.6%
Very important	6	35.3%

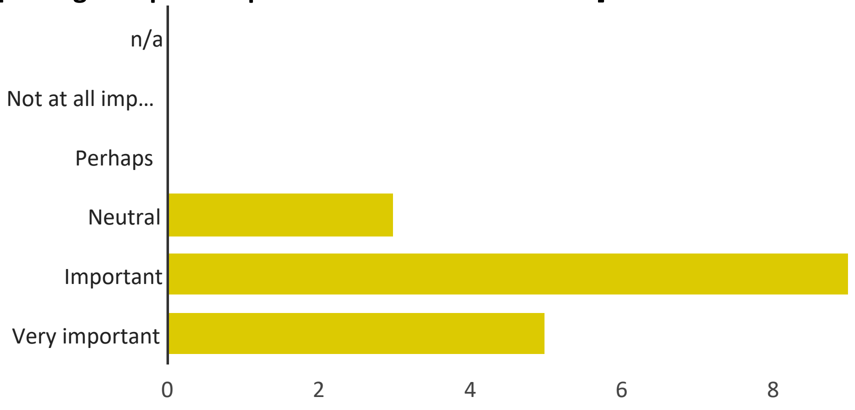
Multi-core
[4. Targeted parallel platforms for the Curriculum]



n/a	1	5.9%
Not at all important	0	0%
Perhaps	0	0%
Neutral	0	0%
Important	3	17.6%
Very important	13	76.5%

Symmetric multiprocessing

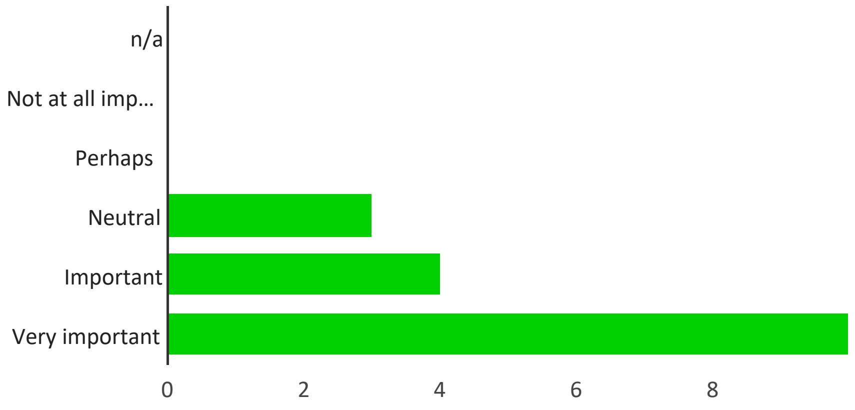
[4. Targeted parallel platforms for the Curriculum]



n/a	0	0%
Not at all important	0	0%
Perhaps	0	0%
Neutral	3	17.6%
Important	9	52.9%
Very important	5	29.4%

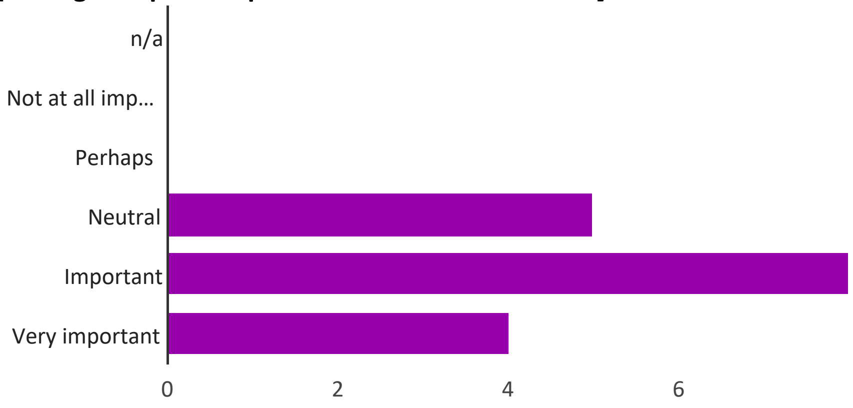
Distributed

[4. Targeted parallel platforms for the Curriculum]



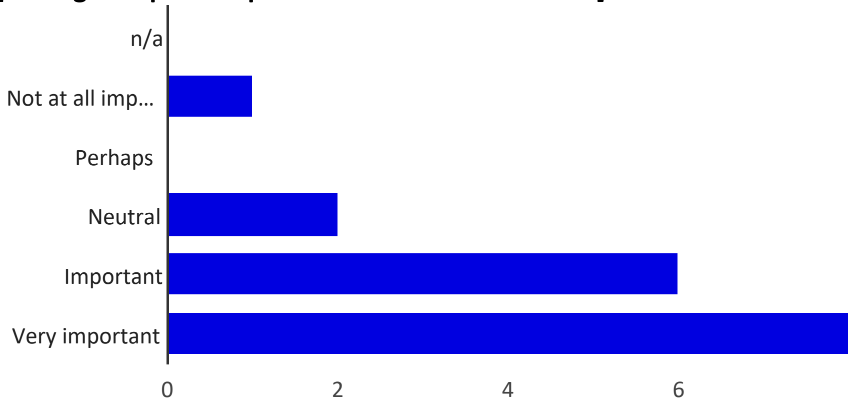
n/a	0	0%
Not at all important	0	0%
Perhaps	0	0%
Neutral	3	17.6%
Important	4	23.5%
Very important	10	58.8%

Cluster/Grid
[4. Targeted parallel platforms for the Curriculum]



n/a	0	0%
Not at all important	0	0%
Perhaps	0	0%
Neutral	5	29.4%
Important	8	47.1%
Very important	4	23.5%

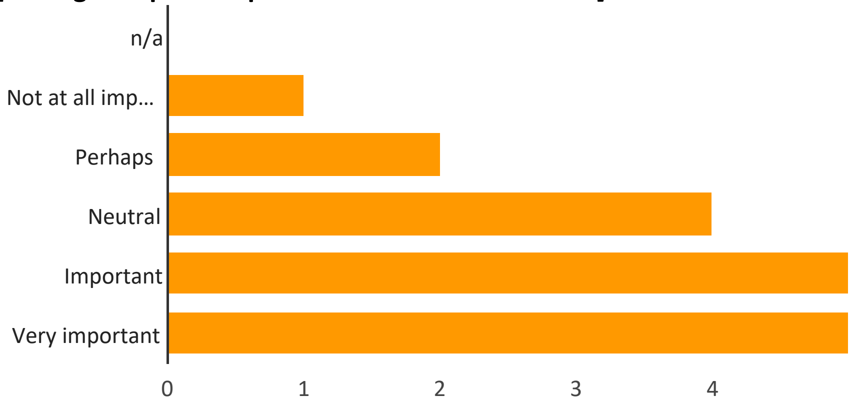
GPGPU
[4. Targeted parallel platforms for the Curriculum]



n/a	0	0%
Not at all important	1	5.9%
Perhaps	0	0%
Neutral	2	11.8%
Important	6	35.3%

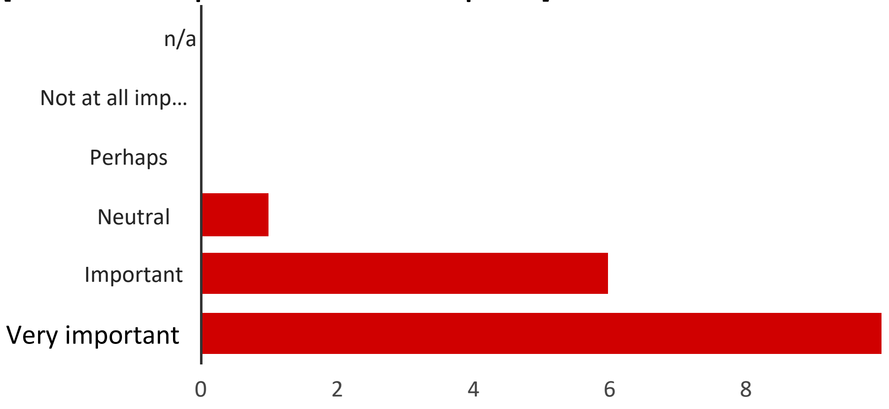
Very important	8	47.1%

Custom (Please specify below)
[4. Targeted parallel platforms for the Curriculum]



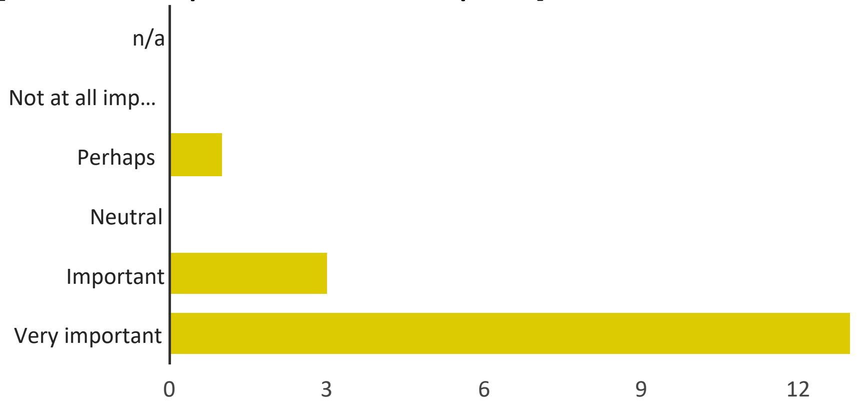
n/a	0	0%	
Not at all important	1	5.9%	
Perhaps	2	11.8%	
Neutral	4	23.5%	
Important	5	29.4%	FPGAs, Accelerators, reconfigurable systems, Heterogeneous SoC, open to customers need
Very important	5	29.4%	Hybrid incorporating conventional CPUs, GPUs and reconfigurable FPGAs for application/domain specifc functional units, Heterogeneous SoC, Manycore accelerators, global shared address space (eg pgas)

Shared memory multiprocessors
[5. Parallel Computer Architecture emphasis]



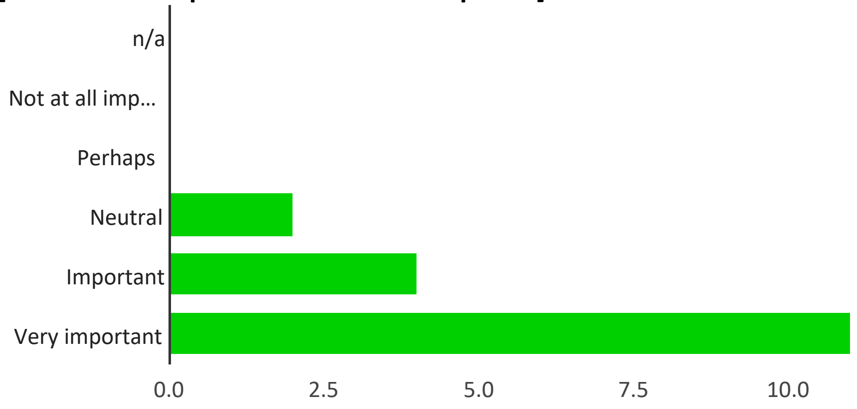
n/a	0	0%
Not at all important	0	0%
Perhaps	0	0%
Neutral	1	5.9%
Important	6	35.3%
Very important	10	58.8%

Multi-core systems
[5. Parallel Computer Architecture emphasis]



n/a	0	0%
Not at all important	0	0%
Perhaps	1	5.9%
Neutral	0	0%
Important	3	17.6%
Very important	13	76.5%

Heterogenous
[5. Parallel Computer Architecture emphasis]

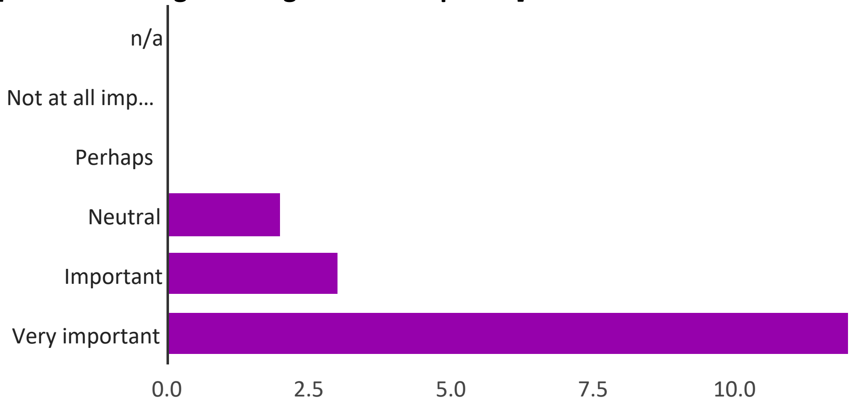


n/a	0	0%
Not at all important	0	0%
Perhaps	0	0%
Neutral	2	11.8%
Important	4	23.5%
Very important	11	64.7%

Other (please specify)

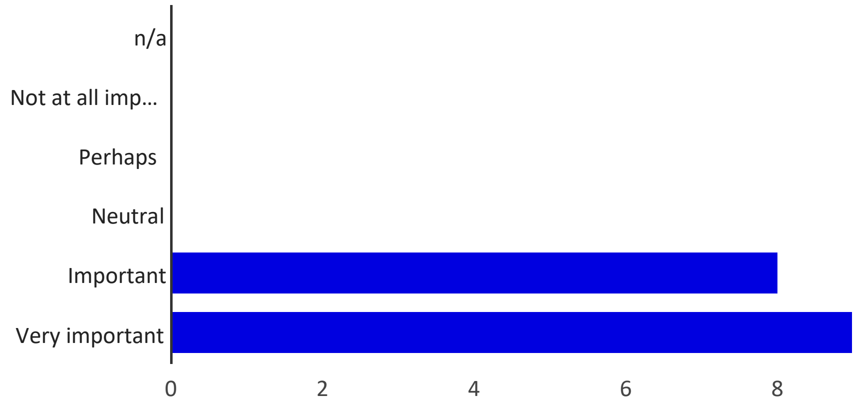
- Heterogeneous should encompass architectures such as ARMs big.LITTLE, CPUs +
- FPGAs + GPUs, mpSOC Embedded multicore
- need to understand the full hierarchy and where the granularity of concurrency moves from shared memory to communications

Shared Memory Model
[6. Parallel Programming Models emphasis]



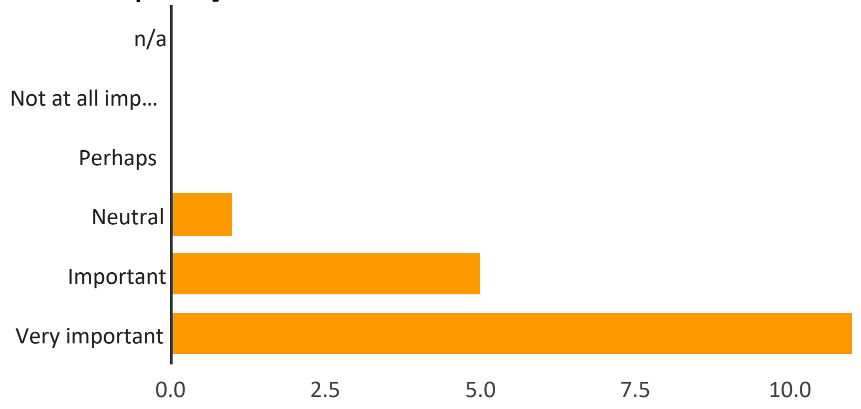
n/a	0	0%
Not at all important	0	0%
Perhaps	0	0%
Neutral	2	11.8%
Important	3	17.6%
Very important	12	70.6%

Threads Model
[6. Parallel Programming Models emphasis]



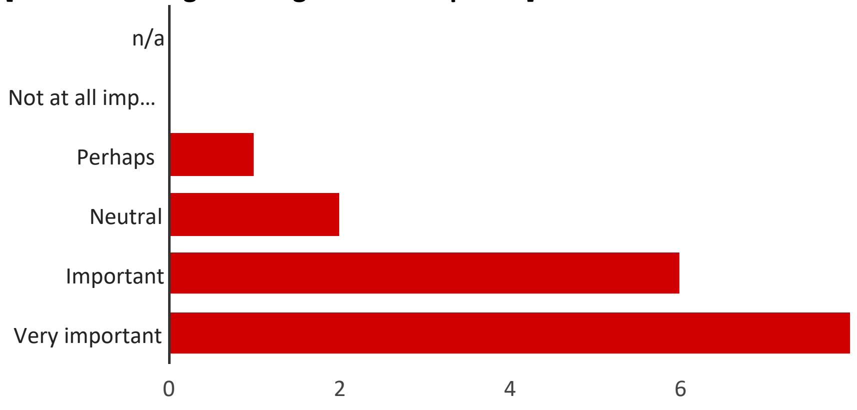
n/a	0	0%
Not at all important	0	0%
Perhaps	0	0%
Neutral	0	0%
Important	8	47.1%
Very important	9	52.9%

Distributed Memory / Message Passing Model [6. Parallel Programming Models emphasis]



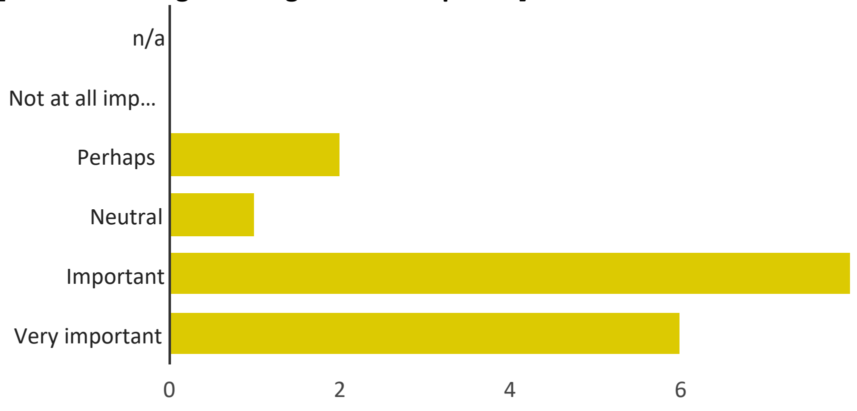
n/a	0	0%
Not at all important	0	0%
Perhaps	0	0%
Neutral	1	5.9%
Important	5	29.4%
Very important	11	64.7%

Partitioned Global Address Space (PGAS)
[6. Parallel Programming Models emphasis]



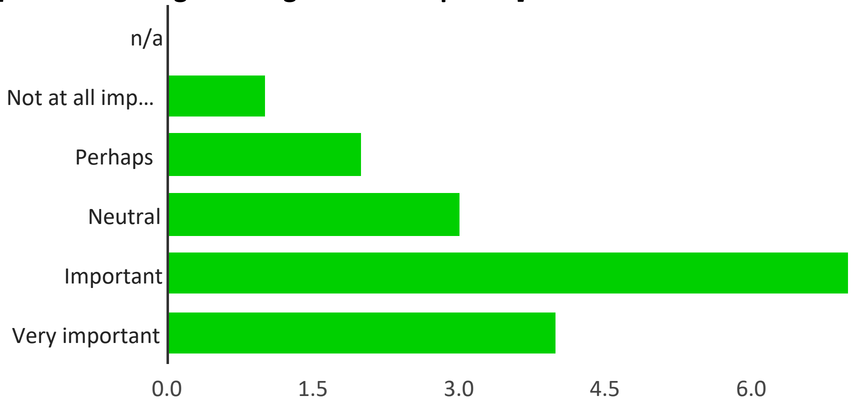
n/a	0	0%
Not at all important	0	0%
Perhaps	1	5.9%
Neutral	5	11.8%
Important	6	35.3%
Very important	8	47.1%

Data Parallel Model (map/reduce, etc)
[6. Parallel Programming Models emphasis]



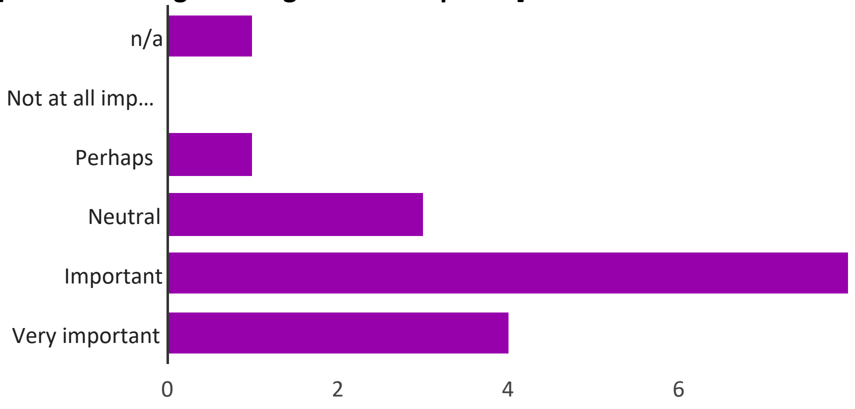
n/a	0	0%
Not at all important	0	0%
Perhaps	2	11.8%
Neutral	1	5.9%
Important	8	47.1%
Very important	6	35.3%

Hybrid Model
[6. Parallel Programming Models emphasis]



n/a	0	0%
Not at all important	1	5.9%
Perhaps	2	11.8%
Neutral	3	17.6%
Important	7	41.2%
Very important	4	23.5%

SPMD
[6. Parallel Programming Models emphasis]

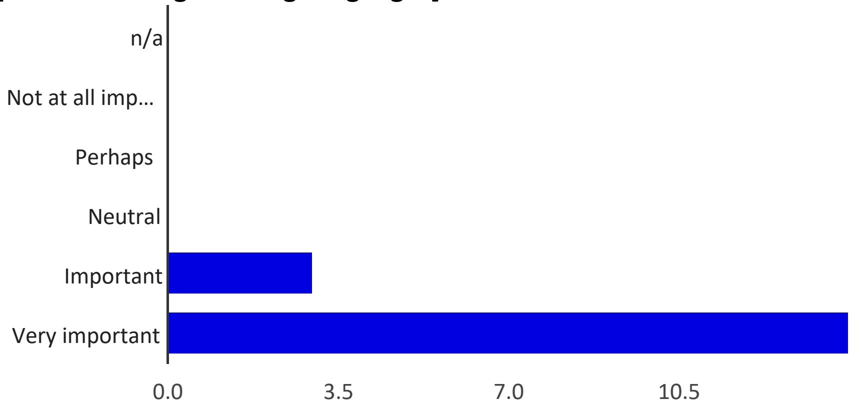


n/a	1	5.9%
Not at all important	0	0%
Perhaps	1	5.9%
Neutral	3	17.6%
Important	8	47.1%
Very important	4	23.5%

Other (please specify)

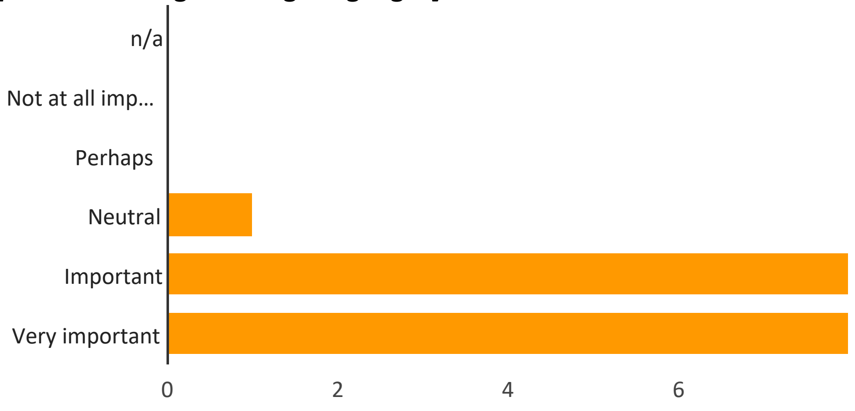
- SIMD

Thread-based (C, C++, OpenMP, etc)
[7. Parallel Programming Languages]



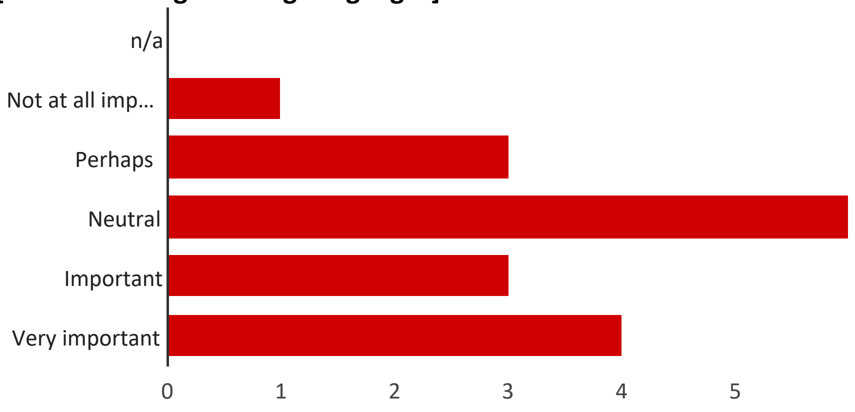
n/a	0	0%
Not at all important	0	0%
Perhaps	0	0%
Neutral	0	0%
Important	3	17.6%
Very important	14	82.4%

OpenCL
[7. Parallel Programming Languages]



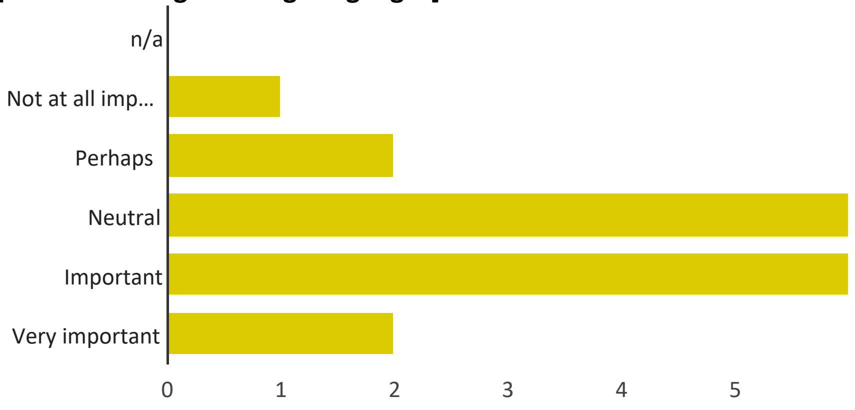
n/a	0	0%
Not at all important	0	0%
Perhaps	0	0%
Neutral	1	5.9%
Important	8	47.1%
Very important	8	47.1%

High-performance FORTRAN
[7. Parallel Programming Languages]



n/a	0	0%
Not at all important	1	5.9%
Perhaps	3	17.6%
Neutral	6	35.3%
Important	3	17.6%
Very important	4	23.5%

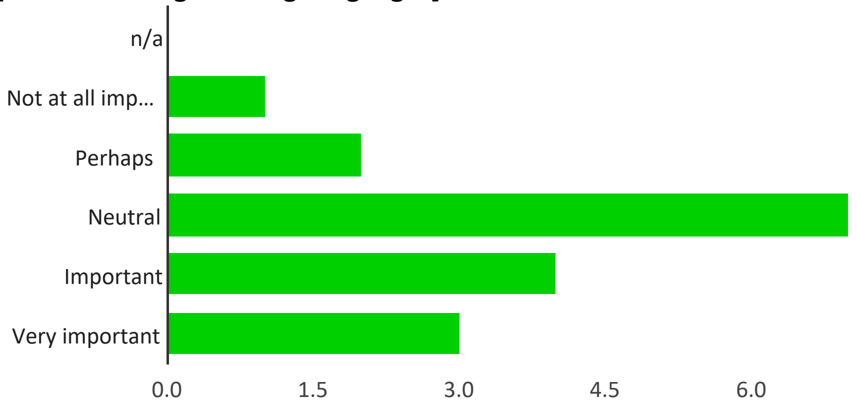
Cilk
[7. Parallel Programming Languages]



n/a	0	0%
Not at all important	1	5.9%
Perhaps	2	11.8%
Neutral	6	35.3%

Important	6	35.3%
Very important	2	11.8%

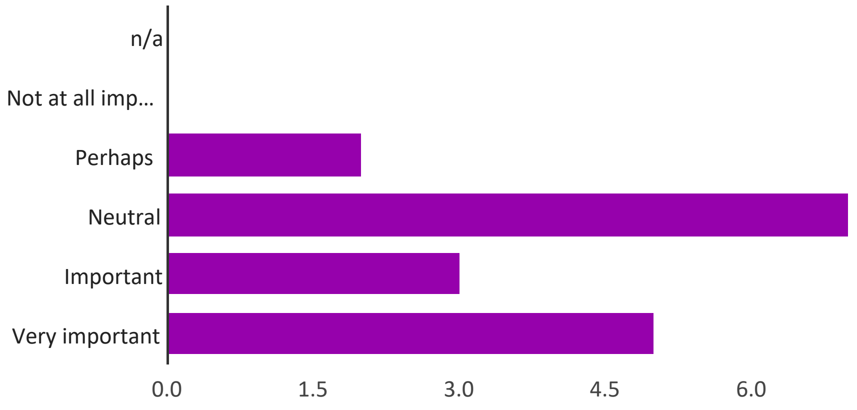
Functional Parallel Programming (Scala/Erlang) [7. Parallel Programming Languages]



n/a	0	0%
Not at all important	1	5.9%
Perhaps	2	11.8%
Neutral	7	41.2%
Important	4	23.5%
Very important	3	17.6%

Java w/ parallel libraries

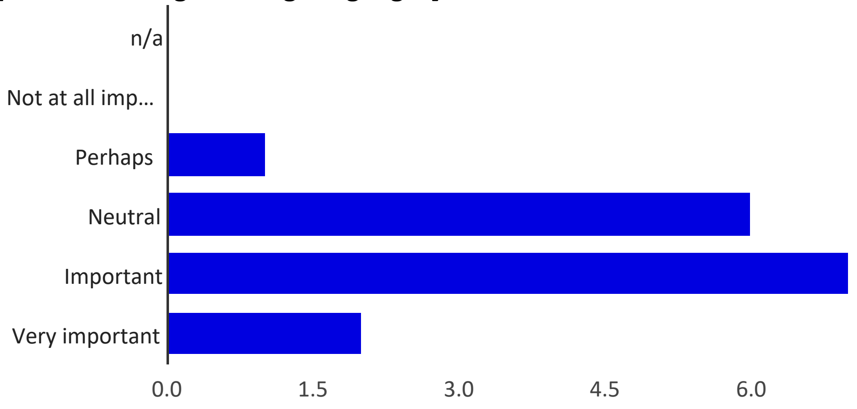
[7. Parallel Programming Languages]



n/a	0	0%
Not at all important	0	0%

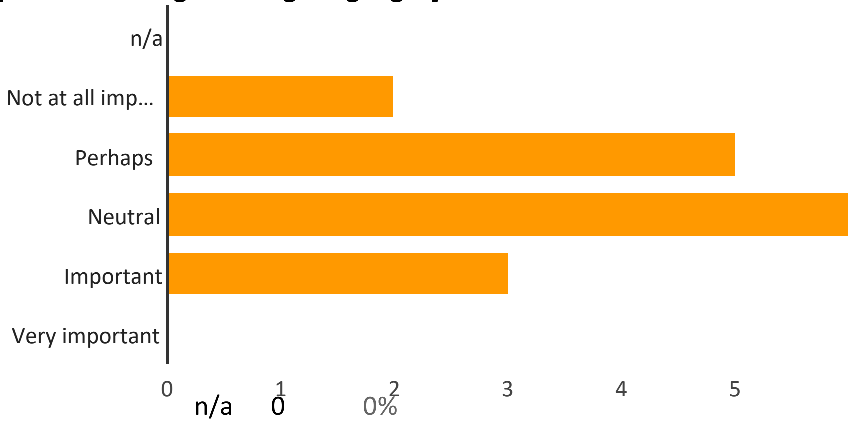
Perhaps	2	11.8%
Neutral	7	41.2%
Important	3	17.6%
Very important	5	29.4%

Threading Building Blocks (TBB)
[7. Parallel Programming Languages]



n/a	0	0%
Not at all important	0	0%
Perhaps	1	6.3%
Neutral	6	37.5%
Important	7	43.8%
Very important	2	12.5%

.NET
[7. Parallel Programming Languages]



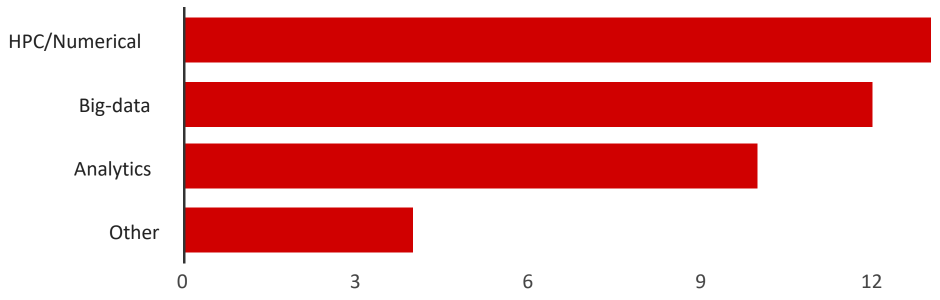
n/a	0	0%
Not at all important	2	12.5%
Perhaps	5	31.3%

Neutral	6	37.5%
Important	3	18.8%
Very important	0	0%

Other (please specify)

- FORTRAN
- CPN
- MPI, Theoretical models

8. Application emphasis:



HPC/Numerical	13	76.5%
Big-data	12	70.6%
Analytics	10	58.8%
Other	4	23.5%

Part B: Detailed catalogue of related courses available

Course name	CS527	Advanced Computer Architecture	Advanced System-on-chip Design: Integrated Parallel Computing Architectures
Brief description, pointer to course and syllabus	http://www.csd.uoc.gr/~hy527/	MOOC under development	http://www.vvz.ethz.ch/Vorlesungsverzeichnis/lerneinheitPre.do?lerneinheitId=91257&semkez=2014S&lang=en
ECTS credit hours	6	7.5	6
Course Enrollment	10	No limit	30
Qualifications Framework Course Levels (Bologna Agreement)	Level 7: Masters/Postgraduate Certificate, Level 8: PhD/Professional Doctorate	Level 7: Masters/Postgraduate Certificate, Level 8: PhD/Professional Doctorate	Level 7: Masters/Postgraduate Certificate
Online/distance learning version available?	No	Yes	No
Pointer to online course material		Will be available fall of 2016	

Can create guest account to on-line resources for evaluation purposes?	Yes	No	Yes
If yes, Contact name	Angelos Bilas	Per Stenstrom	Luca Benini
If yes, Contact email	bilas@ics.forth.gr	pers@chalmers.se	lbenini@iis.ee.ethz.ch

Course name	Parallel Programming	Multicore embedded applications design	Parallel Computer Architecture
Brief description, pointer to course and syllabus	http://www.inf.ed.ac.uk/teaching/courses/ppls/		Introductory Grad course for parallel architectures and basic principles of parallel programming
ECTS credit hours	10	4	7
Course Enrollment	40	20	15
Qualifications Framework Course Levels (Bologna Agreement)	Level 6: Bachelor Degree, Graduate Certificate/Diploma,	Level 6: Bachelor Degree, Graduate Certificate/Diploma,	Level 7: Masters/Postgraduate Certificate
Online/distance learning version available?	No	No	No
Pointer to online course material			
Can create guest account to on-line resources for evaluation purposes?	No	No	No
If yes, Contact name			Dionisios Pnevmatikatos
If yes, Contact email			pnevmati@ece.tuc.gr

Course name	MIHPS: **Master in High Performance Computing, Simulation**	Multi-core Programmierung	Vertiefte Multicore Programmierung
Brief description, pointer to course and syllabus	http://mihps.prism.uvsq.fr/	https://www.informatik.uni-augsburg.de/de/lehrstuehle/sik/lehre/ws/multicore/	not yet available
ECTS credit hours		5	8
Course Enrollment		25	15
Qualifications Framework Course Levels (Bologna Agreement)		Level 6: Bachelor Degree, Graduate Certificate/Diploma,	Level 7: Masters/Postgraduate Certificate
Online/distance learning version available?	No	No	No
Pointer to online course material			

Can create guest account to on-line resources for evaluation purposes?	No	No	No
If yes, Contact name	Nahid Emad, Professeur	Prof. Dr. Theo Ungerer	Prof. Dr. Theo Ungerer
If yes, Contact email		ungerer@informatik.uni-augsburg.de	ungerer@informatik.uni-augsburg.de

Course name	Designing for Parallelism and Future Multi-core Computing	Parallel Programs and their Performance	Introduction to/Intermediate OpenCL
Brief description, pointer to course and syllabus	http://studentnet.cs.manchester.ac.uk/syllabus/?code=COMP60621#learningoutcomes	http://studentnet.cs.manchester.ac.uk/syllabus/?code=COMP60611#learningoutcomes	http://wiki.rac.manchester.ac.uk/community/OpenCL
ECTS credit hours	15	15	7
Course Enrollment	25	25	60
Qualifications Framework Course Levels (Bologna Agreement)	Level 7: Masters/Postgraduate Certificate, Level 8: PhD/Professional Doctorate	Level 7: Masters/Postgraduate Certificate, Level 8: PhD/Professional Doctorate	Level 6: Bachelor Degree, Graduate Certificate/Diploma,, Level 7: Masters/Postgraduate Certificate
Online/distance learning version available?	Yes	No	Yes
Pointer to online course material	http://studentnet.cs.manchester.ac.uk/pgt/COMP60632/		http://wiki.rac.manchester.ac.uk/community/OpenCL
Can create guest account to on-line resources for evaluation purposes?	No	Yes	Yes
If yes, Contact name	Mikel Lujan	Andy Nisbet	Andy Nisbet
If yes, Contact email		Andy.nisbet@manchester.ac.uk	Andy.Nisbet@manchester.ac.uk

Course name	Introduction to/Intermediate CUDA	Python and TAU for High Performance Computing	Intro to OpenMP & Multicore Computing
Brief description, pointer to course and syllabus	https://app.manchester.ac.uk/training/profile.aspx?unitid=6057&parentId=4	https://app.manchester.ac.uk/training/profile.aspx?unitid=6453&parentId=4	https://app.manchester.ac.uk/training/profile.aspx?unitid=4055&parentId=4
ECTS credit hours	7	4	7
Course Enrollment	60	unknown	unknown
Qualifications Framework Course Levels (Bologna Agreement)	Level 7: Masters/Postgraduate Certificate, Level 8: PhD/Professional Doctorate	Level 7: Masters/Postgraduate Certificate, Level 8: PhD/Professional Doctorate	Level 7: Masters/Postgraduate Certificate, Level 8: PhD/Professional Doctorate
Online/distance	No	No	Yes

learning version available?			
Pointer to online course material			http://wiki.rac.manchester.ac.uk/community/OpenMP
Can create guest account to on-line resources for evaluation purposes?	No	No	No
If yes, Contact name	Andy Nisbet	Andy Nisbet	Andy Nisbet
If yes, Contact email	Andy.Nisbet@manchester.ac.uk	Andy.Nisbet@manchester.ac.uk	Andy.Nisbet@manchester.ac.uk

Course name	Advanced OpenMP (in association with PRACE)	Introduction/Intermediate MPI	Introduction to Intel Xeon Phi
Brief description, pointer to course and syllabus	https://events.prace-ri.eu/event/392/	MPI Intermediate	https://app.manchester.ac.uk/training/profile.aspx?unitid=5580&parentId=4
ECTS credit hours	15	6	14
Course Enrollment	unnknown		unknown
Qualifications Framework Course Levels (Bologna Agreement)	Level 7: Masters/Postgraduate Certificate, Level 8: PhD/Professional Doctorate	Level 7: Masters/Postgraduate Certificate, Level 8: PhD/Professional Doctorate	Level 7: Masters/Postgraduate Certificate, Level 8: PhD/Professional Doctorate
Online/distance learning version available?	Yes	Yes	No
Pointer to online course material	http://www.archer.ac.uk/training/course-material/2015/07/advopenmp_manch/	http://wiki.rac.manchester.ac.uk/community/MPI	
Can create guest account to on-line resources for evaluation purposes?	No	No	No
If yes, Contact name	Andy Nisbet	Andy Nisbet	Andy Nisbet
If yes, Contact email	Andy.Nisbet@manchester.ac.uk	Andy.Nisbet@manchester.ac.uk	Andy.Nisbet@manchester.ac.uk

Course name	Operating Systems	Iterative Linear Solvers and Parallelization	Efficient Parallel Programming with GASPI
Brief description, pointer to course and syllabus	http://studentnet.cs.manchester.ac.uk/syllabus/?code=COMP25111#learningoutcomes	http://www.hlrs.de/organization/sos/par/services/training/2016/ITER-S	http://www.hlrs.de/organization/sos/par/services/training/2016/GASPI
ECTS credit hours	40	-	-
Course Enrollment	200	max. 50	max. 30

Qualifications Framework Course Levels (Bologna Agreement)			
Online/distance learning version available?	Yes	Yes	Yes
Pointer to online course material	www.cs.man.ac.uk/	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html
Can create guest account to on-line resources for evaluation purposes?	No	No	No
If yes, Contact name	Andy Nisbet	Dr. Rolf Rabenseifner	Dr. Jutta Oexle
If yes, Contact email	Andy.Nisbet@manchester.ac.uk	rabenseifner@hlrs.de	oexle[at]hlrs.de

Course name	CFD with OpenFOAM®	Introduction to Hybrid Programming in HPC	GPU Programming using CUDA
Brief description, pointer to course and syllabus	http://www.hlrs.de/training/2016/OF1	http://www.hlrs.de/organization/sos/par/services/training/2016/HY-S	http://www.hlrs.de/training/2016/CUDA1
ECTS credit hours	-	-	-
Course Enrollment	max. 50	max. 50	max. 50
Qualifications Framework Course Levels (Bologna Agreement)			
Online/distance learning version available?	Yes	Yes	Yes
Pointer to online course material	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html
Can create guest account to on-line resources for evaluation purposes?	No	No	No
If yes, Contact name	Dr. Rolf Rabenseifner	Dr. Rolf Rabenseifner	Dr. Rolf Rabenseifner
If yes, Contact email	rabenseifner@hlrs.de	rabenseifner[at]hlrs.de	rabenseifner@hlrs.de

Course name	Cluster Workshop	OpenMP GPU Directives for Parallel Accelerated Supercomputers -- an alternative to CUDA from Cray perspective.	Node-level Performance Engineering
Brief description, pointer to course and syllabus	http://www.hlrs.de/organization/sos/par/services/training/2016/CLUSTER	http://www.hlrs.de/organization/sos/par/services/training/2016/XK	http://www.hlrs.de/organization/sos/par/services/training/2016/NLP

ECTS credit hours	-	-	-
Course Enrollment	max. 30	max. 50	max. 50
Qualifications Framework Course Levels (Bologna Agreement)			
Online/distance learning version available?	Yes	Yes	Yes
Pointer to online course material	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html
Can create guest account to on-line resources for evaluation purposes?	No	No	No
If yes, Contact name	Dr. Rolf Rabenseifner	Dr. Rolf Rabenseifner	Dr. Rolf Rabenseifner
If yes, Contact email	rabenseifner[at]hlrs.de	rabenseifner@hlrs.de	rabenseifner[at]hlrs.de

Course name	Fortran for Scientific Computing	Introduction to Computational Fluid Dynamics in High Performance Computing	Cray XC40, Parallel I/O, and Optimization Courses
Brief description, pointer to course and syllabus	http://www.hlrs.de/organization/sos/par/services/training/2016/FTN1	http://www.hlrs.de/organization/sos/par/services/training/2016/CFD-S	http://www.hlrs.de/organization/sos/par/services/training/2016/XC40-1
ECTS credit hours	-	-	-
Course Enrollment	max. 50	max. 50	max. 50
Qualifications Framework Course Levels (Bologna Agreement)			
Online/distance learning version available?	Yes	Yes	Yes
Pointer to online course material	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html
Can create guest account to on-line resources for evaluation purposes?	No	No	No
If yes, Contact name	Dr. Rolf Rabenseifner	Dr. Rolf Rabenseifner	Dr. Rolf Rabenseifner
If yes, Contact email	rabenseifner[at]hlrs.de	rabenseifner[at]hlrs.de	rabenseifner[at]hlrs.de

Course name	Scientific Visualization	Parallel Programming Workshop: Distributed and shared memory parallelization with MPI and OpenMP	CFD with OpenFOAM®

Brief description, pointer to course and syllabus	http://www.hlrs.de/organization/sos/par/services/training/2016/VIS1	http://www.hlrs.de/organization/sos/par/services/training/2016/PAR	http://www.hlrs.de/organization/sos/par/services/training/2016/OF2
ECTS credit hours	-	-	-
Course Enrollment	max. 30	max. 50	max. 50
Qualifications Framework Course Levels (Bologna Agreement)			
Online/distance learning version available?	Yes	Yes	Yes
Pointer to online course material	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html
Can create guest account to on-line resources for evaluation purposes?	No	No	No
If yes, Contact name	Dr. Rolf Rabenseifner	Dr. Rolf Rabenseifner	Dr. Rolf Rabenseifner
If yes, Contact email	rabenseifner[at]hlrs.de	rabenseifner[at]hlrs.de	rabenseifner[at]hlrs.de

Course name	NEC SX-ACE - Vectorization and Optimization	Cray XC40, Parallel I/O, and Optimization Courses	Introduction to Unified Parallel C (UPC) and Co-array Fortran (CAF)
Brief description, pointer to course and syllabus	http://www.hlrs.de/organization/sos/par/services/training/2016/NEC	http://www.hlrs.de/organization/sos/par/services/training/2016/XC40-2	http://www.hlrs.de/organization/sos/par/services/training/2016/UPC1
ECTS credit hours	-	-	-
Course Enrollment	max. 30	max. 50	max. 50
Qualifications Framework Course Levels (Bologna Agreement)			
Online/distance learning version available?	Yes	Yes	Yes
Pointer to online course material	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html
Can create guest account to on-line resources for evaluation purposes?	No	No	No
If yes, Contact name	Dr. Rolf Rabenseifner	Dr. Rolf Rabenseifner	Dr. Rolf Rabenseifner
If yes, Contact email	rabenseifner[at]hlrs.de	rabenseifner[at]hlrs.de	rabenseifner[at]hlrs.de

Course name	Scientific Visualization		
Brief description, pointer to course and syllabus	http://www.hlrs.de/organization/sos/par/services/training/2016/VIS2		
ECTS credit hours	-		
Course Enrollment	max. 30		
Qualifications Framework Course Levels (Bologna Agreement)			
Online/distance learning version available?	Yes		
Pointer to online course material	https://fs.hlrs.de/projects/par/par_prog_ws/practical/README.html		
Can create guest account to on-line resources for evaluation purposes?	No		
If yes, Contact name	Dr. Rolf Rabenseifner		
If yes, Contact email	rabenseifner[at]hlrs.de		