Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2009) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Computational Science, ICCS 2010

CyberInfrastructures of Cyber-Applications-Systems

Frederica Darema*

NSF
Arlington VA, 22230

Abstract

The paper discusses foundational notions and conditions in complex applications modeling and in software frameworks to support advanced CyberInfrastructure environments such as those implied and required for Dynamic Data Driven Applications Systems, as well as other classes of applications, collectively referred to here as Cyber-Applications-Systems. The paper discusses considerations of dynamic invocation of multi-scale models, uncertainty quantification, uncertainty propagation, and dynamic runtime systems support on heterogeneous, distributed, end-to-end dynamically integrated high-end, real-time and instrumentation and control systems.

Keywords: dynamic data driven applications systems; DDDAS; InfoSymbiotic Systems; InfoSymbiotics; multi-scale, multi-modal, multi-data; dynamic multi-scale modeling; uncertainty quantification; uncertainty propagation; cyberinfrastructure; software architectural frameworks; real-time; high-end; instrumentation; control

1. Introduction

Dynamic Data Driven Applications Systems (DDDAS [1]) is an important class of *Cyber-Applications-Systems* [2] [i]. The paper discusses the scope of foundational notions and conditions in complex applications modeling approaches and in implementations and development of software frameworks to support advanced CyberInfrastructure environments, such as those implied and required for emerging classes of applications systems such as Dynamic Data Driven Applications Systems, as well as other classes of complex applications, collectively referred to here as Cyber-Applications-Systems. The DDDAS concept, entailing dynamic integration of computational with instrumentation and control aspects of an application system, implies capabilities that go beyond the standard requirements and considerations in other classes of complex applications such as traditional multi-scale modeling approaches. DDDAS implies dynamic requirements at the application system level as well as dynamic underlying resource management, the later supported through comprehensive software frameworks, referred to here as Cyber-Systems-Software [ii]. In this paper we introduce the term: *InfoSymbiotic* Systems (and *InfoSymbiotics*), to denote systems such as DDDAS, and their capabilities and requirements. The paper discusses considerations of dynamic invocation of multi-scale models, uncertainty quantification and uncertainty propagation in such dynamic applications systems, and dynamic runtime systems support on heterogeneous, distributed, end-to-end dynamically integrated high-end, real-time and instrumentation and control systems.

We have used the term Cyber-Applications-Systems to refer to complex applications systems together with their support environments, for applications representing natural or physical systems, engineered systems, and other

complex man-made systems, such as civil infrastructure systems, business-process models or social-systems models, which complex computational and data support requirements. The new and advanced capabilities discussed here are aimed to address challenges inherent in such systems, namely: complex applications modeling such as multi-scale, multi-modal, multilevel and multiphase modeling, as well as multi-data aspects (multi-*[iii]); applications systems of systems modeling and in dynamic data driven applications systems (DDDAS, SBES[3], and other challenge areas, such as those discussed in the 14 NAE Grand Challenges[4]); addressing the computational and the data requirements of such applications, and ascertaining fidelity of models through uncertainty quantification, validation and verification of models and their associated software programs, and application analytics. As the complexity of applications and that of their support environments increases, other aspects are relevant in supporting these applications and their environments; these include: application composition, application software evolution, maintenance, stewardship, creation of repositories of application models and their associated algorithms, and application programming environments and runtime support.

Together with Cyber-Applications-Systems, Cyber-Systems-Software [iv] represent advanced, end-to-end systems software enabling: Software Infrastructure Frameworks encompassing programming and runtime support environments for Cyber-Applications-Systems, including situations where the applications need to utilize in an optimized way heterogeneous and distributed resources and achieve quality-of-service, as well as conditions where the underlying resources and the applications requirements can change dynamically, as in the case of DDDAS environments.

We refer the reader to [Ref. 1] for more detailed discussion of the infrastructure and frameworks for systems hardware and systems software design, runtime and maintenance planning, and also support development of applications, application portability across platforms and optimized runtime, and transformative transition of such applications to the "exascale" domain; and Compiler Infrastructures for developing advanced compiler systems, such as those enabling dynamic and adaptive mapping of applications under dynamic underlying resources and dynamically changing requirements of applications at runtime.

2. Application Systems of Systems

2.1. Complex, Multi- Applications*

Mathematical and software representations of complex systems entail multiple interoperating components, capable of representing the multiple characteristics and behaviors of such systems, and dynamically extensible and adaptable to either the complex systems behaviors and their changes or changes in their supporting computational (hardware and software) environments. Representing complex systems such as: natural/physical, engineering systems, including computer software and hardware systems, business systems, business process modeling, social systems, etc), often requires ability to represent their various modalities and aspects, including their behaviors and properties as affected by dynamically changing conditions, some of them hard or impossible to specify and predict a-priori.

Such considerations point to representations of complex application systems which consist of collections of models (multi-modal, multi-phase, multi-level, multi-scale in space and time collectively referred to here as multi-*), and which draw their data from multiple sources, instrumentation measurements, other executing models, which need to be architected together in a systems of systems (of applications models).

A number of Workshops, TaskForces, Blue-Ribbon Panels, and other Reports [Refs. 3, 5, 6, 7, 8, and 9] discuss the need, the opportunities and the challenges where understanding observed behaviors and predicting behaviors in a system involves many levels of models and spanning multiple time-scales and/or spatial-scales. Some illustrative examples include: biological systems modeling, ranging from molecular dynamics, to protein folding, to environmental factors affecting the behavior of the dynamics of a biological cell, to an organism, etc., that is: from quantum, to molecular and continuum (or macroscopic) descriptions of systems; e.g. ion hydration, in oil and gas exploration and recovery (discussed in Ref.2) where the scales of models required range from density

functional theory, to molecular dynamics, to nano- and micro-fluidics and micromechanics, to macro-chemistry of large molecules and macro-fluidics, to seismic and EM imaging, to geo-mechanics of rock fracturing, and large scale reservoir modeling; dynamic systems like hurricanes, involve complex atmospheric turbulence models coupled with oceanic water surge models, and multiple observational monitoring data. In addition to multi-* needs, such applications can benefit from new computational paradigms, as discussed in the subsection below.

In multi-* modeling, both from the application modeling and application algorithms point of view, one needs to ensure that appropriate models are included, depending for example for the system configuration that needs to be modeled, so that the application conceptual or mathematical representation reflects the multiple modalities of the application system, such as multi-physics, or multiple dimensional- or time-scales, representing various aspects of the system. In multi-modal, multilevel and/or multi-scale modeling, one needs, for example, to address challenges of how to link models of disparate time and spatial scales; and understand and maintain algorithmic stability across the multi-modal models and the differing time and spatial scales. We discuss support for such compositional capabilities in the Subsection A.4 here, on Application Components and Application Composition Systems; enabling such capabilities is part of the overall CyberInfrastructure frameworks needed for the development and runtime support of the complex multi-* applications discussed in the present report and in [Ref.1].

2.2. New Paradigms and Directions

As has been articulated in the Dynamic Data Driven Applications Systems **(**DDDAS) concept, DDDAS **"***entails the ability to dynamically incorporate additional data into an executing application, and in reverse, the ability of an application to dynamically steer the measurement process"* **[v]**. The approach results into more accurate modeling and ability to speed-up the computation (by augmenting or replacing targeted parts of the computation by the measurement data), thus improving analysis and prediction capabilities of the application model. In addition, application-driven measurement capabilities in DDDAS enable more efficient and effective measurement processes. DDDAS environments entail integration across a range of software systems from high performance computational environments to real-time systems, and a range of hardware and platforms from high-end platforms to sensors and other instruments and real-time data acquisition systems, to "PDAs" and RFIDs. DDDAS is a key concept to improving modeling of systems under dynamic conditions, and is a key concept in architecting and controlling dynamic and heterogeneous resources, including sensor networks, networks of embedded controllers, and other networked resources.

The transformative advances in computational modeling of applications (an in particular those that represent dynamic systems) enabled through the DDDAS concept have been articulated in several forums and demonstrated through advances in a wide set of applications (presented in the ICCS DDDAS Workshop Series [vi] , the DDDAS program solicitation [Ref. **Error! Bookmark not defined.**], many articles on DDDAS based-research **[vii]**, and the SBES Blue-Ribbon Report [Ref. 3]. These references discuss extensively that to enable such capabilities requires multidisciplinary research, and specifically the need for synergistic and systematic collaborations between applications domain researchers with researchers in mathematics and statistics, researchers computer sciences, and researchers involved in the design and implementation of measurement and control (methods, instruments, and other sensors and other control systems). Enabling and supporting DDDAS capabilities entails modeling methodologies and runtime support that go beyond present approaches. For example, dynamically integrating additional and external data, streamed into the executing application, to replace or complement part of the computation, entails and requires for example the ability of the application algorithms (numeric and non-numeric) to be tolerant, and maintain good convergence properties under perturbations from the streamed data. The dynamically incorporated data into the executing application can cause the need to invoke dynamically models representing other scales, behaviors and modalities of the application. This implies not only multi-scale modeling capabilities (discussed in the next sub-section), but dynamic invocation of these models of differing scales and/or differing modalities. Furthermore such dynamic computational requirements imply ability to dynamically acquire underlying support resources and dynamically and optimally map these application components into such resources, and support the ensuing runtime conditions on heterogeneous and distributed

computational, communication, and data-support environments. Likewise in DDDAS environments, uncertainty quantification and uncertainty propagation (discussed in a subsequent subsection) is not only concerning a single model but how uncertainties propagate across dynamically invoked models and how uncertainties in the streamed data can be quantified.

In addition, there are a number of large instrumentation cyberinfrastructures efforts that have been initiated (e.g. those by NSF and cited in Ref. 1) which can benefit from the DDDAS concept. These include newly established instrumentation cyberinfrastructures, as well as a number of earlier efforts as they evolve into the future, enabled through advances in high-end computing, grid computing, and now also cloud computing, sensor systems, etc. Examples include targeted cyberinfrastructure-enabling projects such as: the Network for Earthquake Engineering Simulation (NEES); the Open Science Grid (which includes earlier efforts such as the "international Virtual Data Grid Laboratory – iVDGL"), the Large Hadron Collider (LHC); the Chemistry and Materials Consortium for Advanced Radiation Sources (ChemMatCARS); Energy Recovery Linac (ERL) at CHESS; the Vibrational Spectrometer for Spallation Neutron Source (SNS) at ORNL (VISION); Data Acquisition For Neutron Scattering Experiments (DANSE), and Center for High Resolution Neutron Scattering (CHRNS); the National Ecological Observatory Network (NEON) and the Federation of Environmental Networks (FEON); and the Geosciences Network (GEON); all these and more that may be pursued in the future, provide excellent examples of applications where DDDAS can be used to enhance research productivity and the impact of simulation and measurements enabled by these infrastructure projects.

To exemplify the CyberInfrastructure frameworks needed for the development and runtime support of the DDDAS applications and their environments here are summarized some of the research and technology development challenges and opportunities [Refs 1 and 2]. In DDDAS implementations, an application/simulation must be able to accept data at execution time and be dynamically steered by such dynamic data inputs. This requires research advances in application models that: describe the application system at different levels of detail and modalities; are able to dynamically invoke appropriate models as needed by the dynamically injected data into the application; and include interfaces of applications to measurements and other data systems. **Application Measurement Systems and Methods** include improvements and innovations in instrumentation platforms, and improvements in the means and methods for collecting data, focusing in a region of relevant measurements, controlling sampling rates, multiplexing, and determining the architecture of networked sensor assemblies and other measurement systems. Advances in **Mathematical and Statistical Algorithms** include creating algorithms with stable and robust convergence properties under perturbations induced by dynamic data inputs: algorithmic stability under dynamic data injection/streaming; algorithmic tolerance to data perturbations; multiple scales and model reduction; enhanced asynchronous algorithms with stable convergence properties. Advances in **Systems Software runtime support and infrastructures** to support the execution of applications whose computational systems resource requirements are dynamically dependent on dynamic data inputs, and include: dynamic selection at runtime of application components embodying algorithms suitable for the kinds of solution approaches depending on the streamed data, and depending on the underlying resources, dynamic workflow driven systems, coupling domain specific workflow for interoperation with computational software, general execution workflow, software engineering techniques. **Software Infrastructures** and other systems software (OS, data-management systems and other middleware) services to address the "real time" coupling of data and computations across a wide area heterogeneous dynamic resources and associated adaptations while ensuring application correctness and consistency, and satisfying time and policy constraints. Specific features include the ability to process large volume, high rate data from different sources including sensor systems, archives, other computations, instruments, etc.; interfaces to physical devices (including sensor systems and actuators), and dynamic data management requirements. Also support is needed for accessing and visualizing at runtime, computational, measured and sensed data. In addition the new environments considered here require integrated frameworks for accessing grid resources that support research exploration, workflow capture and replay, and a dynamic services oriented architecture, with standards protocols. All these, and also as articulated in [Refs 1 and 2], also need to be integrated in into comprehensive ***software frameworks*** to support **DDDAS CyberInfrastructures**, and into **Community Testbeds.** A number of DDDAS research efforts have started developing cyberinfrastructure software frameworks; such efforts can evolve to robust prototype versions and tools for a broader set of efforts.

In the next Section are discussed other important considerations in computational modeling, namely ascertaining validity and fidelity of the models, by understanding and addressing the impact of uncertainties in models and data, validation and verification of models, etc. In DDDAS environments these challenges, of modeling uncertainty, propagation of uncertainty, etc, they are further augmented as the application itself changes dynamically at execution time. DDDAS environments spur the need for advances in these areas.

3. Uncertainty, Validation and Verification, Application Analytics, and Application Driven Analytics

One of the most important tenets of modeling is the ability to analyze systems to understand the observed and predict the yet unobserved. In other words one of the objectives of modeling is to be a "predictive science". To fulfill such goals, assessing the fidelity and accuracy of application models is very important. This is necessary, not only to be able to accurately analyze and understand the modeled system, but also to be able to make accurate predictions of not yet observed characteristics, properties, and behaviors of a system, and to discover new phenomena in natural and physical systems, predict the range of capabilities and predict new capabilities in engineered systems, predict the systems behaviors under these new capabilities, and analyze such behaviors in hard to test or in rare extreme situations. Understanding the range of applicability of a model is thus very important for achieving such objectives.

3.1. Uncertainty, Validation and Verification

The fidelity of a model is affected considerably not only by the accuracy by which the model represents a system, but also by the uncertainty in the data that are used to drive the model, and how these uncertainties propagate as the simulation proceeds. Much work has been devoted in evaluating the impact of these sources of uncertainty as well as uncertainty propagation, including faster methods for computing the propagation of the uncertainty bounds. There are many methods of calculating uncertainty, including faster polynomial based methods. Also related to that is estimation of the fidelity of computation in cases of faulty or incomplete data [viii]. Given the multi-* systems and other emerging dynamic systems (like DDDAS) discussed here, efforts in estimation of uncertainties remain important areas.

Validation is the process of determining the accuracy by which a model (or the collection of models in the case of multi-modal, multi-level, multi-scale modeling) represents the actual system, the accuracy by which it can characterize and describe the actual system, and/or the accuracy by which the model can predict behaviors of a system. **Verification** is the accuracy by which a computational representation (computer program) of the mathematical model implements the mathematical model. Validation asks: are the right equations used and solved mathematically? Verification asks: does the computer program solve these equations correctly? [note: this is a somewhat different definition that the one given in Ref. 3].

Verification of the model (that is of the computer program) involves software engineering methods, for testing and ensuring correctness of execution, and detection of various programming and execution errors (program bugs). While software engineering methods have been effective in small and custom designed applications, for the large, complex, and compute- and data-intensive applications of interest here, software engineering methods are rather slow and inadequate to provide verification for entire programs

With current approaches, for large applications models, both validation and verification of models and their associated computer programs, is done by checking the fidelity of the model (and its associated computer program) against some predetermined expectations either dictated by data from actual instantiations of the system. Thus, validation of the model entails checking its ability against points or regions of the solution space. Typically is not possible to cover all the solution space, the models are validated, against "important" test cases, rather than exhaustively; in fact, typically it is a small subset of the solution space of the problem.

The opportunities for research here are in developing more comprehensive approaches of verification and validation not only of individual models, but also of the complex collections of models, such as those that are encountered in multi-modal and multi-scale applications modeling (hat is in application systems of systems). Similar challenges and opportunities concern effects of uncertainties in modeling and the fidelity of results, as discussed earlier in reference to multi-*.

3.2. Application Analytics and Application Driven Analytics

Related to considerations about multi-scale, multi-level, multi-modal modeling, uncertainty, validation and verification, are the technology and research areas relating to **Application Analytics, and Application Driven Analytics**. Some key challenges and opportunities are discussed next.

Application analytics is the study allowing to understand the range of validity of the application models (the individual models and complex assemblies of such models, as in the case of multi-*). Application analytics includes quantification of the quality of input application data, their uncertainties, or inconsistent and incomplete data, and understanding the range and coverage of tests in the validation of the application models and the coverage of testing in the verification of the associated application software.

In a related fashion, we consider in this report application analytics as connected to **Application Driven Analytics** of the actual system represented by the application. In this context the application model is used to understand a range of behaviors and characteristics of the system, and also the bounds of such behaviors and characteristics; for example understanding the range of stability of an engineered system under certain conditions and how a (complex) system will behave when perturbed beyond its design points. Fidelity of such application-driven analytics of the system depends on the understanding of the range of validity of the application model(s) involved (as defined and assessed through application analytics).

Another perspective of applying analytics is in **Application Driven Analytics** of computer systems. An approach for testing the capabilities of systems' software and of the underlying hardware platforms is to create test cases driven by known applications (whose models have been validated, and their respective application software has been verified). Through this approach one can create a sampling of scenarios and quantification of quality attributes of the characteristics and behavior of the said systems software and the hardware platforms. **Benchmarking** is a case example of application-driven analytics that allows characterizing the capabilities of a software or hardware system driven from a range of selected applications (benchmarking set). Often it is desirable to test, through such benchmarking methods, the capabilities of entire computational environment stack, provided by the hardware platforms together with the supporting systems software. As discuss in [Ref. 1] a methodology to perform such an analysis of software and hardware requires systematic approaches, such as modeling and analysis software frameworks encompassing multi-level, multi-modal, multi-scale application software, systems software and hardware models.

4. Application Components and Application Composition Systems

From the application software and application execution point of view we need to address aspects of the requirements and capabilities for composing such applications, the need for assists and tools that can enable the composition of the application at the development time (for example knowledge based recommender systems). Other aspects to be considered in the application composition are for example application models and algorithms appropriate for the data sets considered and the characteristics of the underlying platforms on which the application will execute.

In these cases, as we discuss later-on the CyberSystemsSoftware section, one also needs to consider the ability of application composition dynamically at execution time, in the cases where the underlying resources change and/or the requirements of the application change as the application executes. Approaches discussed there are high-level programming tools (e.g. visual programming or script–driven high-level user interfaces for user directed assists), as

well as advanced problem solving environments, recommender systems, using archival high-level descriptions of characteristics of such components, their data, and the underlying platforms architectural characteristics, and knowledge based approaches, to enable optimized composition of applications, at application development stage and also dynamic composition at execution time, in response to changing underlying resources and/or changing requirements of the application (for example in executing models of multiple scales).

Furthermore, in the case of DDDAS environments, scientific and engineering simulations of complex physical phenomena, entail combining dynamically (and opportunistically, as needed) computations, instrumentation measurements (archival and real-time data), and frequently involve dynamic invocation of application models and algorithms. For example in a given application (representing a physical, engineering, biological, economic system) models of different levels or modalities may be dynamically invoked, as dictated by the dynamic data inputs, and the ability to dynamically compose and couple different components is a common theme in DDDAS applications. Another related common theme is the need to support dynamic assimilation of data and from varying numbers and classes of sensors. The need for and ability to dynamically compose simulation components, to support dynamic choice between different simulation models and on-demand real-time sensor data assimilation requirements pose substantial systems software challenges. Approaches to dynamic composition are likely to involve metadata management schemes, development of schemes for supporting semantic functional and performance oriented data service interfaces along with workflow scheduling and management schemes.

The challenges here relate and cross-over the issues discussed in the section on uncertainty quantification, model verification and validation. The discussion of the challenges there applies not only to the individual models, but also the dynamic composition of such models, in the "application systems of systems" discussed here.

5. Application Software Evolution and Maintenance

Software for applications modeling evolves, because of several broad factors, such as: changes in the application modeling approaches, changes in application software and the support software environments, and changes in the hardware platforms:

- changes in the application modeling approaches include: improved understanding of the theoretical principles or the fundamental concepts about the system, leading to improvements in the application models, new modeling paradigms, new modeling methods and advances in the fundamental algorithms involved in the modeling, need to meet new analysis requirements, need to incorporate additional factors (such as multi-modal, multilevel, multi-scale capabilities), couple the given model(s) with other applications' models (for example: in fire modeling, couple radiative-heat transfer models with air turbulence models), interface models with additional data sources (differing data models);
- changes in application software, include correcting errors in the application program and other software faults such as software performance and reliability deficiencies; adapt the application software to operate in new execution environments, like new programming paradigms, hardware platforms, runtime systems, operating systems, data management and I/O systems; and overall modifying the application software architecture, rewriting parts of the application to improve software structure, to make the application software more efficient and maintainable. {e.g. evolving the software framework… }
- changes in the support software environments (like OS, compilers, I/O, DBMS systems, etc) and of course changes in the hardware architecture (processing node architecture, interconnects architecture, size of memory in any of the memory hierarchies, peripherals, or other hardware resources), all these they often precipitate needs to change (or evolve) the application software, especially when one wants also the application to execute with optimized performance and QoS.

All these challenges argue for more systematic methods for fundamental computer sciences research and synergistic collaboration between computer-science researchers and computational scientists to develop software engineering approaches and tools, driven-by and accommodating the needs of large, complex, and dynamic applications referenced here.

The open software approach is an important and relatively recent approach, which does have impact in approaches on software evolution and maintenance. Of course open software has other broader impacts (such allowing multiple stakeholders to use the software developed), but here we discuss open software in the context of evolution and maintenance, and as complementary to other approaches addressing the objectives discussed above and in reference to its role for addressing gaps in these efforts. For example, by making the software available to the broader community, an application model, or an application component, or an algorithm, not only the contribution is used and leveraged by others, but also by having more users deploy this software, then it can be tested more extensively, thus providing more comprehensive testing, and validation and verification of the application models and their associated software.

In summary, it is important to provide support of the full software life-cycle in the ecosystem of people, software, data and hardware, and in the context of software frameworks spanning the application, the systems software and application services layers and bridging to the hardware layers.

6. Repositories of Applications Models

The increasing complexity of applications and the associated software has motivated for sometime now the need to create libraries of application models and algorithms. It is the desire to leverage rather than duplicate work in application modeling and algorithmic research, and in software development. Examples there-of include numerical libraries (like LINPACK, LAPACK, EISPACK, etc), software PETC, and numerous application packages (e.g. GAUSSIAN, GAMESS, AMBER, CHARMM, NAMD, GROMOS, LAMMPS, etc, just to name some packages in computational chemistry area and molecular dynamics methods; many other science and engineering areas also have corresponding applications packages, the list being too long to include all here).

In addition, several efforts have aimed in the past at creating problem solving environments where knowledge based systems can "recommend" appropriate models, algorithms, and other software components (e.g. PSEs, the "recommender system" projects [10]). While these efforts have provided considerable help to application developers, they have not reached the level of capabilities that are needed for the kinds of advanced dynamic application systems and environments that are encountered today and envisioned in the future. Such present and future application systems require use of "knowledge-based recommender" systems (as fostered by the NGS program) to automate the mapping and optimized runtime of applications (as discussed in more detail in the systems software section below), and provide the support that is envisioned as needed by emerging application environments (such as the DDDAS discussed earlier-on). Also such "knowledge-based" recommender systems are needed to support dynamic Application Composition Systems capabilities (ideas discussed earlier-on in this report, and fostered with the NGS and DDDAS Programs, and also discussed in the CyberSystemsSoftware section below).

In addition to application models, one needs the repositories to include information on the range of validity of the models, both in terms of the application cases that such models may apply, and also in terms of the data sets and data sizes that the models are suitable, the underlying platforms and data-sets to test the models, and ability to include new data sets in the panoply of model testing. Again knowledge-based approaches are also applicable here.

Furthermore, the kinds of model repositories discussed here will need to include data sets for validation of the models and verification of their associated software, and also capture information over time, as the models evolve and as their associated software methods evolve, and also as the underlying hardware platforms evolve. For example, one needs to capture not only data and metadata about the data, but also include the input of developers' and other experts who have been involved in the development of the models and the software (their comments, remarks and other information).

7. Application Programming Environments

Modern applications consist of multiple inter-operating compute- and data-intensive components. To obtain accurate models and simulations, or to deliver real-time results, the applications need to execute on high-performance globally distributed and high-end petaflops-classes (and now hexaflops- and beyond) computing platforms. At the same time these applications need to achieve high-efficiency and QoS when executing on such platforms.

The present methods of building applications result in applications that are designed for a given platform. When the underlying platform changes often significant parts of the application need to be rewritten for the new platform(s). This is costly and limiting: the resulting applications cannot automatically move to the new platform; the applications cannot be distributed to run concurrently on the old and the new platforms; the applications cannot be dynamically partitioned across globally distributed platform assemblies, map dynamically across such platforms as the resource availability changes and exploit such platform assemblies with quality-of-service. Similar obstacles exist when the problem size changes and the application needs, for example, to be repartitioned and remapped for the bigger problem size. Today's technology mostly relies on considerable and laborious hand tuning. Over the last decade, efforts to enable more effective use of distributed computing platforms have been launched. These efforts articulated the need for automating the process of distributing and mapping the application across such platforms, as well as optimizing the mapping of the application on a given high-end platform (e.g.: NGS 1998-2004 [11]; IBM Autonomic Computing Initiative in 2001 [12]).

The advent of multi-core based platforms, will accentuate the referenced problems, to even higher levels of complexity, and exascale architectures and exascale-class applications augment the complexity of these kinds of requirements and challenges. Furthermore, applications environments like DDDAS they also pose a number of unique system software and support requirements. These kinds of applications involve an element of adaptive real time response (hard- or soft-, or both, depending on the control task or application phase), that is, they require seamlessly integrated environments, from “the real-time to the high-end”. Discussion of computational models and runtime support for DDDAS environments are provided in [13, 14, and Ref. 2]. Systems software (also referred to as middleware [ix]) used to support DDDAS therefore needs to be able to address aspects of performance prediction, performance negotiation, and performance guarantees. Furthermore, DDDAS environments require system level quality of service guarantees. DDDAS applications typically require multiple computational inputs within sometimes predictable, and sometimes unpredictable, periods of time. The timescales differ from application to application, and possibly for different stages and tasks of a given application. For example, some applications may require millisecond level responses while others require a range of responses in seconds, hours, days or weeks depending on the nature of the control task. In all of these cases, DDDAS software stacks need to be able to support predictable temporal response to such varying characteristics. In such environments it becomes imperative to augment and more systematically support efforts such as in [Ref. 2].

8. Summary

This paper has addressed has addressed considerations in advanced complex applications modelling, such as multi-scale modelling and uncertainty quantification, and in particular in the context of dynamically integrated computational, instrumentation and control systems, such as in DDDAS, and in the context of creating cyberinfrastructures supporting such environments as encompassed in Cyber-Applications-Systems and Cyber-Systems-Software frameworks entailed in these environments.

References

1. DDDAS: www.cise.nsf.gov/dddas and www.dddas.org

2. F. Darema, Monograph submitted for publication by Springer; the term Cyber-Applications-Systems introduced ibid; and prior to that in the NSF internal reports: F. Darema: Report on New Opportunities for OCI: *CyberIfrastructures of Cyber-Applications-Systems & Cyber-Systems-Software, October 2009, and F. Darema Report on* Industrial Partnerships in Cyberinfrastructure *("Innovation through CyberInfrastructure Excellence"- ICIE*)
3. SBES-Oden: http://www.nsf.gov/pubs/reports/sbes_final_report.pdf
4. NAE: http://www.engineeringchallenges.org
5. Frederica Darema-Transformative Research Report [March 2008]; included in Ref.1
6. http://www.exascale.org/mediawiki/images/a/a7/Messina-doeexa.pdf - Workshops Series (organized by Paul Messina): Climate, November 6-7, 2008; HEP, December 9-11, 2008; Nuclear Physics, January 26-28, 2009; Fusion Energy Sciences, March 18-20, 2009; Nuclear Energy, May 11-12, 2009; BES, August 13-15, 2009; Biology, August 17-19, 2009; NNSA, October 6-8, 2009
7. http://computing.ornl.gov/workshops/town_hall/reports/mod_sim.pdf; Modeling at the exascale for Energy and the Environment (Horst Simon, Thomas Zacharia and Rick Stevens)
8. www.exascale.org – IEST: International Exascale Software Project (led by Jack Dongarra) with a series of workshops starting at SC08, Nov 18, 2008, and with three recent ones in Santa Fe, NM USA, April 7-8, 2008; Paris, France, June 28-29, 2009; Tsukuba, Japan, October 19-21, 2009
9. http://www.wtec.org/sbes/ and SBES-Global: http://www.wtec.org/sbes/SBES-GlobalFinalReport.pdf - International Assessment of Research and Development in
Simulation-based Engineering and Science
10. Rice-Houstis-Ramakrishnan, etal first "recommender project" in 1997 and PYTHIA-II (2000), and follow-on work by Ramakrishnan; and POEMS project (NGS) James Browne et.al. (2000)
11. NGS: http://www.nsf.gov/pubs/2001/nsf01147/nsf01147.htm (and previous 1998 -)
12. http://www.research.ibm.com/autonomic
13. Frederica Darema, Characterizing Dynamic Data Driven Applications Systems (DDDAS) in Terms of a Computational Model, Lecture Notes in Computer Science (LNCS), Volume 5545/2009, May 2009; and Journal of Algorithms and Computational Technology (JACT)
14. F. Darema**,** Grid Computing and Beyond: The Context of Dynamic Data Driven. Applications Systems, *(Invited Paper),*Proceedings of the IEEE, Special Issue on Grid Computing, *Edited by M. Parashar and C. A. Lee,* Vol. 3, No. 3, March 2005

---

[i] Term introduced by the author in Ref.2
[ii] Term introduced by the author in Ref.2
[iii] "multi-*" denotes "multi-model" and "multi-data" applications; the term "multi-model" is used in this report as encompassing: multi-modal, multi-phase, multi-level, multi-scale (also refer to subsequent sections and discussions in Ref.1)
[iv] Term introduced by the author in Ref.2
[v] *"DDDAS is a paradigm whereby application/simulations and their measurements become a symbiotic feedback control system, enabling applications that adapt intelligently to evolving conditions, and that infer new knowledge in ways that are not predetermined by startup parameters."* In addition through the notion of dynamic control of measurement processes dictated by the application at runtime, entails capabilities for dynamic and optimized management of heterogeneous measurement and other resources.
[vi] DDDAS Workshops: http://www.nsf.gov/cise/cns/dddas/2006_Workshop/index.jsp and
http://www.nsf.gov/cise/cns/dddas/index_wkshp.jsp
[vii] DDDAS Workshop Series at the International Conference on Computational Sciences (yearly 2003-2009, …)
[viii] See also Oil&Gas Exploration&Production application example in [Ref.1]
[ix] Although in principle middleware (which originally started as software for the networking layer) is a subset of Systems Software