EDISON Science Gateway: A Cyber-Environment for Domain-Neutral Scientific Computing

Hoon RYU†, Jung-Lok YU†, Duseok JIN†, Jun-Hyung LEE†, Dukyun NAM†, Jong suk LEE†, Kumwon CHO†, Nonmembers, Hee-Jung BYUN††, Member, and Okhwan BYEON††, Nonmember

SUMMARY We discuss a new high performance computing service (HPCS) platform that has been developed to provide domain-neutral computing service under the governmental support from “EDucation-research Integration through Simulation On the Net” (EDISON) project. With a first focus on technical features, we not only present in-depth explanations of the implementation details, but also describe the strengths of the EDISON platform against the successful nanoHUB.org gateway. To validate the performance and utility of the platform, we provide benchmarking results for the resource virtualization framework, and prove the stability and promptness of the EDISON platform in processing simulation requests by analyzing several statistical datasets obtained from a three-month trial service in the initiative area of computational nanoelectronics. We firmly believe that this work provides a good opportunity for understanding the science gateway project ongoing for the first time in Republic of Korea, and that the technical details presented here can be served as an useful guideline for any potential designs of HPCS platforms.

key words: science gateway, high performance computing service (HPCS), computational science, technology computer-aided design (TCAD) simulations

1. Introduction

High performance computing (HPC) services have become general and available to public with a remarkable progress in computing power, networks, storages and online environments that have driven during the last decade [1], [2]. As a consequence, “computational science” itself has become established as an important keyword for researches and educations in various fields of science and engineering, where researchers have not only solved advanced problems that are quite computationally expensive, but successfully demonstrated their results have impacts that are critical enough to guide the design processes needed by experimentalists [3]–[5].

Needs for Science gateway: Computational scientists who build their own technology computer-aided design (TCAD) softwares running in parallel on HPC resources, commonly uses command line interface (CLI) clients such as Terminal as a gateway to access clusters. To many more people who want to research or get educated using existing TCAD softwares but are not familiar of command instructions, however, CLI clients may not be the most efficient way to access TCAD services as they first needs to become familiar with the set of instructions to configure and run TCAD softwares in clusters. The issue of providing domain scientists with “easier” TCAD services has thus motivated the strong needs for a Science Gateway coupled to graphical user interface (GUI) clients such as web portals.

The concept of the science gateway can be defined as a HPC service (HPCS) platform that integrates a community-specific set of TCAD softwares, other applications to support simulation services (e.g., visualizations), and collections of related dataset coupled to user-friendly clients to provide an easy access to HPC-integrated resources [6], [7]. Several HPCS platforms have been designed to support online simulations in various fields of computational science [8]–[12]. Of these, nanoHUB.org project carried out under the support from US National Science Foundation (NSF)[8]–[10] should be one of successful examples that represents the needs for science gateways providing online simulation service via GUI clients since it has supported an annual user-base that now exceeds 230K focusing on the online simulations of nanoelectronics.

There is no doubt that the above-mentioned HPCS platforms including nanoHUB.org project are leading science gateways as well as providing public services for online TCAD simulations. They are, however, dedicated for simulations in a particular domain area (e.g., nanoelectronics, disasters and chemistry etc). Moreover, all the backend resources integrated to the above-mentioned platforms are located in US such that the service quality may not be good in some (especially Asia-pacific) area due to the networking issue, although the strong network should be the most critical component in maintaining of the service quality for TCAD simulations that typically need to process large datasets.

Goals and impacts of this work - EDISON project: Based on the well-known strong IT infrastructure in Republic of Korea, we have recently launched “EDucation-research Integration through Simulation On the Net” (EDI-SON) project under the financial support of the Ministry of Education, Science and Technology of Republic of Korea, where the main objective of the project is to construct an application-domain neutral HPCS-hub of TCAD softwares that can not only provide a qualified service of online simulations, but be easily accessible by researchers, students and developers in Republic of Korea as well as in Asia-pacific
The main objective of this paperwork is to discuss the details of the EDISON HPCS open platform, which has been developed and released with an trial service area on computational nanoelectronics. Especially we demonstrate the practicability and strength of the EDISON platform by addressing following issues: (1) “Why is the EDISON platform suitable to support domain-neutral TCAD software?”, (2) “What are the strong points of the EDISON platform from a viewpoint of the service management, especially compared to the leading science gateway - nanoHUB.org?”, and (3) “How good is the performance when the system is applied to a real-time service?”. We believe that this work provides a good opportunity for understanding the science gateway project ongoing for the first time in Republic of Korea, and that the technical details discussed here are solid enough to be served as an useful guideline for whom want to start a new implementation of HPCS platforms from scratch.

2. EDISON Platform: Implementation Details

The purpose of the EDISON platform is to manage domain-neutral TCAD softwares and related contents by establishing a convergence environment for simulation services using cyber-infrastructure. The three critical components that a HPCS platform should satisfy to meet the above-mentioned purpose can be summarized as follows:

- **Generality**: The platform must be able to manage various simulation workflows to support a wide range of TCAD software in computational science.
- **Expandability**: The platform must maintain neutrality to computing resources in which TCAD software will run. That is, the platform must provide an abstract interface that is synchronous with any new resource environments.
- **Openness**: The platform must provide an open, web-standard interface to accommodate various user-friendly clients such as web-portals, stand-alone applications, and mobile environments.

Figure 1 provides an overview of the EDISON open platform that has been developed to satisfy the above-mentioned requirements. The platform broadly consists of the middleware stack, the application service framework, and the portal service framework. The EDISON middleware allows the management of TCAD software metadata, the management of simulation datasets, the management of simulation history, and synchronization of heterogeneous (physical and virtual) computing resources. Based on a web-standard REST (Representational State Transfer) ful
interface [13], EDISON application service framework includes the user authentication, simulation workflows, metadata query of TCAD software, storage, and statistical analysis. The EDISON portal service framework offers a set of web-based GUI clients that are specialized for the simulation workflow of each TCAD software.

Of all the components described in the platform hierarchy (Fig. 1), the native web-based GUI client (portal service framework), the Science Appstore service (application framework), and virtualized computing resource/job management framework (middleware stack) are the three key features of the EDISON platform, which eventually provide the efficiency in management of simulation requests as well as the flexibility in supporting various types of the simulation workflow. In next subsections, we discuss these key features further in detail.

2.1 Native Web-Based Interface for Simulations

To become useful for the open computing service, a HPCS platform should not only enable researchers from geographically distributed locations to share their in-house TCAD software and related datasets, but also guarantee that any users can run shared applications without buying a licenses for all toolboxes. Moreover, the platform should provide a set of user-friendly GUI such that researchers/users can perform the sharing/running process easily. Without loss of generality, a web-based problem-solving environment should be the most suitable solution to satisfy the above-mentioned motivations since it not only maximizes the accessibility via browsers, but can be also easily implemented using various open frameworks.

In nanoHUB.org [8], users can control the input parameters and visualize the results of simulations via the web-interfaces that are based on the virtual networking computing (VNC) technology. The VNC-based interface, however, is well known not to be generally suitable for the public service environment since it introduces a substantial overhead that results in a slow and sometimes unreliable graphics protocol when used simultaneously by many users in a wide area network [14]. To avoid the overhead in supporting many users as much as possible, therefore, the EDISON platform allows users to access the service via a web-native GUI that are developed using a server-side scripting language (PHP) [15]. Figure 2(a) shows the native web-interface via which users can control a set of simulation parameters in the EDISON platform.

Another strong point of our native web-interface, is that it can be easily incorporated with the web-standard technologies for visualizations such as HTML5 Canvas, Web Graphics Library (WebGL) and Scalable Vector Graphics (SVG) [16]–[19]. Figure 2(b) shows the visualization interfaces of simulation results that are implemented using WebGL, which supports high-quality visual effects directly in browsers with lesser overhead compared to the VNC-coupled visualizations.

2.2 Science Appstore Framework

In general, the simulation environment is comprised of various components such as preprocessors (e.g., control of simulation inputs), TCAD software, postprocessors (e.g., visualization scheme), computing resources, and job managers. It is however hard to establish an domain-neutral simulation environment since the data schema and interfaces of these components have not been standardized. Existing simulation environments have been dependent on specific types of TCAD software, thus only supporting limited application fields [8]–[12]. With the “Science Appstore” framework that can be understood as a set of metadata schema for the above-mentioned components, the EDISON platform provides the simulation environment that can accommodate a greater variety of TCAD software in diverse fields of computational science.

Metadata schema for TCAD software: For the service of domain-neutral applications, we have carefully examined the workflow patterns of various TCAD software in the fields of computational fluid dynamics, nanoelectronics, chemistry, structural dynamics and medicine, and have
Fig. 3 A conceptual diagram representing the XSD set of TCAD software metadata: The hierarchy and cardinality of standardized metadata schema are shown. The XML-based schema is highly document-centric, and is particularly strong in expandability as the addition/deletion of new elements/attributes are quite flexible.

established a schema of 131 components of software metadata. The 78 elements and 53 attributes are again further classified to a total of six sections with a consideration of interoperations among the components of the simulation environment, where each section composes the XML schema definition (XSD) [20] of the standardized metadata set. Details of the XSD are described as follows, and Fig. 3 illustrates the hierarchy and cardinality of the metadata schema.

- **Identification Section**: This section contains the information needed to identify the TCAD software (the title, version, developer and affiliation) as well as the information needed to help users understand the softwares (the description, features and screenshots of simulation details).
- **Code Section**: This section contains the information needed to create automated job commands (the name and path of the executable file, and the path where result files are stored) as well as the information needed to install and run the software (the programming language, compiler and static libraries required to link).
- **Parameter Section**: This section contains the information needed to create interfaces that allow users to control the set of simulation parameters (the names, data types, descriptions and default values of control parameters).
- **Category Section**: This section contains the information needed to set the simulation workflows (the preprocessor and postprocessor) as well as the information with which users can easily search their preferred TCAD software on the web-portal (the geometries, applications and problem categories of simulations).
- **Additional Section**: This section contains the information that supports the validity of TCAD software (the list of related technical reports, research papers and other contents) as well as the information of users (majors, grades and purposes of portal users).
- **System Section**: This section contains the information with which the administrator can manage softwares being served on the HPCS platform (the version control, usage frequency and status of service).

The Rapid APPlication infrastrucTURE (Rappture) [21], an interface for running TCAD simulations that is currently being utilized in nanoHUB.org, also manages the software metadata based on the XML schema. While the Rappture focuses on describing the input and output metadata of each TCAD software, the Science Appstore framework manages the metadata in a broader manner, categorizing the dataset into workflow components, i.e., inputs/outputs of the TCAD software and types of the preprocessor/postprocessor required by each TCAD software, such that simulations of diverse workflow patterns can be supported flexibly.

Storage of software metadata and API set: Fig. 4(a) illustrates the structure of the Science Appstore framework that is used to store and manage domain-neutral TCAD software. The metadata repository has been developed using a Not-only SQL (NoSQL) database (DB) class [22] that is well-known to be good to preserve unstructured (arbitrary)
datasets. We have especially employed MongoDB to implement the storage system [23], [24], which allows the independent and expandable synchronization between simulation components without constraints on the schema structure yet still supporting the preservation of the document-centric data schema.

The APIs for the management of the software metadata are available as a RESTful interface. The interface has been implemented with Node.js [25], which is a server-side JavaScript designed for implementation of scalable internet applications, notably web-servers. The APIs are written using event-driven, asynchronous inputs and outputs (I/O) to minimize overhead and maximize scalability. The list of the RESTful APIs needed to operate the repository is shown in Fig. 4 (b), which basically provides CRUD (create/read/update/delete) and Query functions to manage metadata of specific TCAD software, pre/postprocessor, and configuration. In detail, the API set can be categorized as follows:

- **Solver**: APIs in this group can be utilized to deploy and manage the metadata of the specific TCAD software. It provides functions to CRUD the software metadata, as well as to search the software based on the user-query. APIs have been designed to provide the system administrators and (TCAD software) developers with the access to storages.

- **Pre/Post-Processor**: APIs in this group provides functions to CRUD the metadata of pre/postprocessors, and to retrieve pre/postprocessors coupled to the specific TCAD software based on the query.

- **Configuration**: APIs in this group provides functions to acquire the metadata of additional configurations (e.g., library packages) needed to run the software.

2.3 Virtualized Resource and Job Management Framework

Resource virtualization should be the one of most essential components for HPCS platforms. The ubiquitous availability of the virtualization technology brings the efficient resource management for public computing services such that one of the benefits is, for example, that it enables the support of multiple hardware architectures and application environments with limited types and numbers of physical machines, providing the versatility of supporting various TCAD applications that work on various application environments [26]. Another component critical for the open computing service is the job management framework via which the system administrators can manage computing resources by monitoring jobs requested by users. In this subsection, we discuss the virtualized computing resources and job management framework of the EDISON platform.

**Structure and functionalities**: The virtualized computing resource/job management framework of the EDISON platform has been developed upon the following design principles:

<table>
<thead>
<tr>
<th>User</th>
<th>Host</th>
<th>VM</th>
<th>VC</th>
<th>Simulation / Job</th>
<th>Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>User Management</strong></td>
<td><strong>Host Management</strong></td>
<td><strong>Virtual Machine</strong></td>
<td><strong>Virtual Network</strong></td>
<td><strong>Virtual Block</strong></td>
<td><strong>Volume Abstraction</strong></td>
</tr>
<tr>
<td><strong>User Management</strong></td>
<td><strong>Host Management</strong></td>
<td><strong>Virtual Machine</strong></td>
<td><strong>Virtual Network</strong></td>
<td><strong>Virtual Block</strong></td>
<td><strong>Volume Abstraction</strong></td>
</tr>
<tr>
<td><strong>User Management</strong></td>
<td><strong>Host Management</strong></td>
<td><strong>Virtual Machine</strong></td>
<td><strong>Virtual Network</strong></td>
<td><strong>Virtual Block</strong></td>
<td><strong>Volume Abstraction</strong></td>
</tr>
<tr>
<td><strong>User Management</strong></td>
<td><strong>Host Management</strong></td>
<td><strong>Virtual Machine</strong></td>
<td><strong>Virtual Network</strong></td>
<td><strong>Virtual Block</strong></td>
<td><strong>Volume Abstraction</strong></td>
</tr>
<tr>
<td><strong>User Management</strong></td>
<td><strong>Host Management</strong></td>
<td><strong>Virtual Machine</strong></td>
<td><strong>Virtual Network</strong></td>
<td><strong>Virtual Block</strong></td>
<td><strong>Volume Abstraction</strong></td>
</tr>
<tr>
<td><strong>User Management</strong></td>
<td><strong>Host Management</strong></td>
<td><strong>Virtual Machine</strong></td>
<td><strong>Virtual Network</strong></td>
<td><strong>Virtual Block</strong></td>
<td><strong>Volume Abstraction</strong></td>
</tr>
<tr>
<td><strong>User Management</strong></td>
<td><strong>Host Management</strong></td>
<td><strong>Virtual Machine</strong></td>
<td><strong>Virtual Network</strong></td>
<td><strong>Virtual Block</strong></td>
<td><strong>Volume Abstraction</strong></td>
</tr>
<tr>
<td><strong>User Management</strong></td>
<td><strong>Host Management</strong></td>
<td><strong>Virtual Machine</strong></td>
<td><strong>Virtual Network</strong></td>
<td><strong>Virtual Block</strong></td>
<td><strong>Volume Abstraction</strong></td>
</tr>
<tr>
<td><strong>User Management</strong></td>
<td><strong>Host Management</strong></td>
<td><strong>Virtual Machine</strong></td>
<td><strong>Virtual Network</strong></td>
<td><strong>Virtual Block</strong></td>
<td><strong>Volume Abstraction</strong></td>
</tr>
</tbody>
</table>

Figure 5 illustrates the hierarchy of the virtualized computing resource/job management framework. The framework consists of the three layers - the abstraction, core-framework, and web-service. The bottommost abstraction layer supports the interoperations among various functional environments such as the authentication, virtualization, resource virtualization, managements of simulation jobs and storage devices. The central layer is the core-framework that provides the actual functionalities required for services such as the user management, (physical/virtual) server provisioning, and simulation/job requests. Finally, the web-service layer supports web-standard RESTful interface, via which the administrator can access and manage the virtualized resources and simulation jobs easily.

The abstraction layer of the framework provides the abstraction interface for the user authentication and managements of the virtualization platform, simulation jobs and storage devices, and thus eventually enable the interoperations between these functionalities and the pre-constructed infrastructure of the server farm and HPC storages. The authentication interface provides the local DB and Lightweight Directory Access Protocol (LDAP, [27]) plug-ins. For on-demand virtual machines (VMs) provisioning, the Xen [28] plug-in, which is built on top of OpenNebula [29] cloud management toolkit, is available on a trial basis. OpenPBS [30] and Network File System (NFS) are supported as plug-ins for the management of simulation jobs and storage devices, respectively.

In the core framework, information of user, physical
server (host), VM, VM image, virtual network, storage and simulation job are managed as Plain Old Java Objects (POJOs) coupled to Hibernate. The simulation jobs, can be summarized as follows:

- **User management/authentication**: The system administrator is able to register and delete the user information. General users can obtain the authentication through the login/logout interface, and can access other service APIs using the token provided if the authentication is successful. The framework uses a HTTP(S) basic authentication mechanism to authenticate users and for delegation of authority.
- **Physical server (host) management**: To perform simulation jobs requested by users, virtual machines/clusters must be provisioned and registered in the shared pool. The framework provides APIs to register/delete physical servers where virtual machines/clusters will be created. Access to the set of APIs in this category is granted to the system administrators only.
- **VM provisioning/management**: APIs in this group are used for provisioning virtual machines/clusters to physical servers that are registered by administrators. Being accessible by both administrators and general users, the API set enables the request of provisioning with information of machine specifications such as a number of processors and size of memory etc. It also provides users with the access to detailed information of provisioned VMs, and allows the VM owner to submit requests to resume/suspend the machines.
- **File I/O**: Generally TCAD simulations not only require multiple input data, but also produce multiple result data upon the completion of jobs. APIs in this group is used to build interfaces via which users can control simulation inputs and access results produced when simulations are finished.
- **Simulation management**: The framework interprets a “simulation” as an virtual parent object, which has a subset of individual jobs that become the child objects, i.e., a simulation for a parameter study consists a “simulation” as an virtual parent object, which has a subset of individual jobs that become the child objects, i.e., a simulation for a parameter study consists a subset of individual jobs that become the child objects. APIs in this group is used to build interfaces via which users can control simulation inputs and access results produced when simulations are finished.
- **Job management**: After creating a simulation object as a virtual parent, users can manage individual jobs, e.g., the specific jobs of interest can be individually controlled simulation inputs and access results produced when simulations are finished.

### APIs for resource and job management: The framework provides a set of 42 RESTful APIs where a full list is shown in Fig. 6. The web-standard interface currently available for the management of computing resources and simulation jobs can be summarized as follows:

#### User Management/Authentication

<table>
<thead>
<tr>
<th>API</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST /api/user/login</td>
<td>POST /api/user/logout</td>
</tr>
<tr>
<td>GET /api/user/logout</td>
<td>GET /api/user/current</td>
</tr>
<tr>
<td>GET /api/user/current</td>
<td>POST /api/user/persistence</td>
</tr>
<tr>
<td>GET /api/user/modify</td>
<td>GET /api/user/login/register</td>
</tr>
<tr>
<td>POST /api/user/register</td>
<td>POST /api/user/login/reset</td>
</tr>
<tr>
<td>PUT /api/user/login/reset</td>
<td>POST /api/user/login/resetpass</td>
</tr>
<tr>
<td>POST /api/user/login/resetpass</td>
<td>PUT /api/user/login/resetpass</td>
</tr>
<tr>
<td>POST /api/user/login/resetpass</td>
<td>POST /api/user/login/resetpass</td>
</tr>
<tr>
<td>POST /api/user/login/resetpass</td>
<td></td>
</tr>
</tbody>
</table>

#### File I/O

<table>
<thead>
<tr>
<th>API</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST /api/file/upload</td>
<td>POST /api/file/download</td>
</tr>
<tr>
<td>GET /api/file/delete</td>
<td>GET /api/file/read</td>
</tr>
<tr>
<td>GET /api/file/write</td>
<td>GET /api/file/modify</td>
</tr>
<tr>
<td>GET /api/file/modify</td>
<td>GET /api/file/modify</td>
</tr>
<tr>
<td>DELETE /api/file/modify</td>
<td>DELETE /api/file/modify</td>
</tr>
<tr>
<td>GET /api/file/modify</td>
<td>GET /api/file/modify</td>
</tr>
</tbody>
</table>

#### Simulation Management

<table>
<thead>
<tr>
<th>API</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST /api/simulation/submit</td>
<td>POST /api/simulation/cancel</td>
</tr>
<tr>
<td>GET /api/simulation/complete</td>
<td>GET /api/simulation/cancel</td>
</tr>
<tr>
<td>GET /api/simulation/complete</td>
<td>GET /api/simulation/complete</td>
</tr>
<tr>
<td>PUT /api/simulation/complete</td>
<td>PUT /api/simulation/complete</td>
</tr>
<tr>
<td>PUT /api/simulation/complete</td>
<td>PUT /api/simulation/complete</td>
</tr>
<tr>
<td>DELETE /api/simulation/complete</td>
<td>DELETE /api/simulation/complete</td>
</tr>
</tbody>
</table>

#### Job Management

<table>
<thead>
<tr>
<th>API</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST /api/job/submit</td>
<td>POST /api/job/cancel</td>
</tr>
<tr>
<td>GET /api/job/submit</td>
<td>GET /api/job/cancel</td>
</tr>
<tr>
<td>GET /api/job/submit</td>
<td>GET /api/job/cancel</td>
</tr>
<tr>
<td>PUT /api/job/submit</td>
<td>PUT /api/job/cancel</td>
</tr>
<tr>
<td>DELETE /api/job/submit</td>
<td>DELETE /api/job/submit</td>
</tr>
</tbody>
</table>

### 3. Results and Discussion: EDISON Nanophysics

To benchmark the utility and hear users’ experience for the improvement of the currently developed EDISON HPCS platform, we have performed a trial service for online simulations during about 3 months, focusing on the initiative service area of the computational nanoelectronics (physics).

A total of 272 users have joined the trial service, performing a total of 4,220 simulation jobs during the service period (see Table 1 for the service outline.) 8 nodes coupled to 1G network, where each node has two 6-core 2.4GHz Intel Xeons (E5650) and 96GB memory, are provided as physical resources (96 cores). Each physical core is then mapped to a set of two VMs that run on CentOS 5.6 with 2 vCPUs and 4GB memory. Xen [33] 4.0.1 Hypervisor is employed as a VM monitor (VMM).

Next subsections discuss the practicality and utility of the EDISON HPCS platform in both a qualitative and quantitative manner. We not only present detailed descriptions of the service environment that has been supported via the EDISON_Nanophysics portal (http://nano.edison.re.kr) during the service period, but also demonstrate the performance of the framework for the resource virtualization and job management when coupled to the real TCAD applications.
Web-interfaces for online simulation services: Users can perform and access simulations easily via user-friendly GUIs for the following 4 steps - (1) Interfaces for searching and selecting TCAD software at users’ preference, (2) an interface for input controls for the selected software, (3) an interface for monitoring the status of requested jobs, and (4) interfaces for accessing the simulation results. All the details in input parameters, supplemental descriptions, workflow patterns and pre/postprocessing of TCAD software should be provided as a set of metadata from developers.

Table 1  
EDISON Nanophysics trial service: Overview.

<table>
<thead>
<tr>
<th>Objective</th>
<th>Service Period</th>
<th>Summary of Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>In-class</td>
<td>2012-Oct-15 ~</td>
<td>168 Users,</td>
</tr>
<tr>
<td>Utilization</td>
<td>2012-Nov-30 (47 days)</td>
<td>2,165 Simulation Jobs</td>
</tr>
<tr>
<td>EDISON</td>
<td>2013-Mar-04 ~</td>
<td>104 Users,</td>
</tr>
<tr>
<td>Competition</td>
<td>2013-Apr-19 (47 days)</td>
<td>2,055 Simulation Jobs</td>
</tr>
</tbody>
</table>

*Utilized in 7 classes in following 5 domestic institutes: Korea Advanced Institute of Science and Technology (KAIST), Sookmyung Women’s Univ., Sejong Univ., Seoul National Univ. (SNU), and Korea Univ.

*EDISON Competition in Nanophysics (electronics) has been held under the support of National Research Foundation of Korea (http://www.nrf.re.kr) to motivate students to perform their own research based on what learned in classrooms.

In nanoelectronics for service.

3.1 Service Environment

Web-interface for online simulations: User-friendly web-interface is one of the critical features that a qualified service environment for online simulations should have. Figure 7 describes the set of web-interfaces that are presented in the EDISON Nanophysics portal to support an “easy” online service for TCAD simulations. To be more specific, the EDISON Nanophysics portal provides the set of GUIs via which users can perform and access TCAD simulations according to the four steps as follows:

- **Selection of a TCAD software**: Once the administrator register the software in the Science Appstore framework with a complete set of software metadata and executables that work correctly, users can easily select their preferred applications via the web-interface which not only shows the list of available TCAD software (STEP 1.1 in Fig. 7), but also provides detailed information related to each TCAD software (STEP 1.2 in Fig. 7).

- **Control of simulation parameters**: Upon the selection of preferred software, users can control the set of input parameters to perform meaningful simulations (STEP 2 in Fig. 7). The interface supports direct inputs of parameter values as well as upload of files needed as preconditions of a simulation. If preprocessors are needed, they can be included in the simulation workflow at the moment of the registration. Brief descriptions for parameters are also available such that users can double-check any potential mistakes while they are working on the simulation setup. Detailed information of the input control (parameter names, types, descriptions, and preprocessors) must be presented as a part of the metadata set by the software developers.

- **Monitoring of submitted jobs**: Once the job is submitted to the backend resource, users can monitoring the status of requested simulation jobs. The monitoring interface (STEP 3 in Fig. 7) provides 4 types of the job status (QUEUED/RUNNING/FAIL/SUCCESS). Jobs that are queued or running can be cancelled at users’ preference.

- **Interpretation of simulation results**: For successfully completed jobs, users can either visualize in a predefined way, or directly download the simulation results. Currently, the visualization is supported in two ways; (1) a in-house tool developed using WebGL [17] for
Table 2  EDISON_Nanophysics trial service: The six applications mostly popularly utilized.

<table>
<thead>
<tr>
<th>Title</th>
<th>Descriptions</th>
<th>Platforms</th>
</tr>
</thead>
<tbody>
<tr>
<td>pnjuncLAB</td>
<td>Solves the electrostatics of 2-terminal P/N junction diodes [35]; Useful to educate semiconductor fundamentals for undergraduates in electrical engineering.</td>
<td>C++, LAPACK [36].</td>
</tr>
<tr>
<td>pwLAB</td>
<td>Solves the quantum confinement of single particle in 1D finite potential wells [37]; Useful to educate core basics in quantum mechanics for undergraduates in physics.</td>
<td>C++, LAPACK, ARPACK [38].</td>
</tr>
<tr>
<td>collisionLAB</td>
<td>Solves the elastic collisions between two objects; Useful to educate the basics in physics for freshmen students in majors of science and engineering.</td>
<td>Fortran (F90 family).</td>
</tr>
<tr>
<td>emesLAB</td>
<td>Solves the electronic structure of 3D finite nanoscale devices with an effective mass framework [39]; Useful in researches on nanoelectronics modeling (material properties); Support large-scale computation running in parallel.</td>
<td>C++/MPI [40], LAPACK, Parallel ARPACK.</td>
</tr>
<tr>
<td>nanowireFET</td>
<td>Solves the 3D quantum transport problems using the Non-equilibrium Green’s Function (NEGF) [41]; Useful in researches of nanoelectronics modeling (current-voltage properties of nanoscale devices); Runs in parallel.</td>
<td>C++/MPI, LAPACK, PETSc [42].</td>
</tr>
</tbody>
</table>

1D data visualizations (STEP 4.1 in Fig. 7), and (2) Paraview [34] open source toolkit for 2D/3D data visualizations (STEP 4.2 in Fig. 7). While Paraview supports the 1D visualization, allowing users to take line-cuts of 2D/3D data, it involves relatively large resource overhead due to the data-rendering engine. For TCAD softwares dedicated to 1D problems where the data-rendering is not critically needed, the visualization can be done using our in-house tool with less overhead.

**TCAD applications available in EDISON_Nanophysics:**

During the service period, 15 applications that are implemented and run in a linux environment, have been served for online simulations supporting educations of core basics and researches in semiconductor engineering fundamentals, quantum mechanics, and electronic structure and quantum transport in advanced nanoscale devices. Details of some applications that have been popularly utilized, can be found from the summary given in Table 2. We note that 10 more applications will become available in the portal prior to the full service of EDISON_Nanophysics that will be launched in near future.

3.2 Performance: EDISON Platform

Since an HPCS platform indeed consists of various components as discussed in the previous section and illustrated in Fig. 1, it should be in principle difficult to define a reasonable set of performance indicators, especially when we focus too much on every technical component only. In this subsection, therefore, we would like to discuss the performance of the EDISON platform focusing on the two viewpoints - resource virtualization and job management, which are the important factors tightly coupled to the cost and quality of the service.

**Virtual machines:** As addressed the resource virtualization is critical, since it serves as a key factor enabling the efficient management of the computing resource and thus could result in a reduction of the cost required to maintain the service. To understand the performance of virtualized computing resources that have been utilized during the trial service period, the execution time of the NAS Parallel Benchmarks (NPB) [43], [44] workloads has been benchmarked against physical resource with a problem size of class C. The four NPB workloads are described as follows (for elaborated descriptions, see Refs. [41] and [42]):

- **EP (Embarrassingly Parallel):** Generate a set of independent Gaussian random variables using the Marsaglia polar method.
- **MG (Multi Grid):** Approximate the solution to a three-dimensional discrete Poisson equation using the V-cycle multigrid method.
- **FT (Fast Fourier Transform):** Solve a three-dimensional partial differential equation (PDE) using the fast Fourier transform.
- **LU (Lower-Upper symmetric Gauss-Siedel):** Solve nonlinear PDEs using the algorithm involving a symmetric successive over-relaxation solver kernel.

The performance of VMs that have been provided for TCAD simulations is demonstrated Fig.8, where we expressed the execution time of test workloads (EP/MG/FT/LU) measured in VMs as a ratio to that measured in physical nodes. The VM performance for EP-type workloads, where the communication ratio among CPUs becomes minimal, turns out to be almost native, i.e., the execution times in VMs are almost identical to that in physical nodes. This, which readers might think is trivial at a first glance, indicates one thing important from a viewpoint of the service, that the resource virtualization we employed is indeed efficient for a simultaneous processing of multiple requests for serial jobs (simulations running on a single CPU) that do not communicate one another at all. Degradation in performance is, however, observed to increase for workloads of larger communication ratio such that FT and LU-type workloads, when run in 32 virtualized cores, take a 2.2 and 2.5 times longer wall-time than in 32 physical cores, respectively.

The experimental result on the Xen-virtualization overhead is shown in Fig. 9, where we expressed the CPU over-
Fig. 8 Performance of virtualized computing resources: VMs become almost native for the EP-type workload indicating that the resource virtualization has been indeed quite successful in managing serial job requests with just 96 physical cores. Degradation in performance, however, is observed as the communication ratio increase (MG<FT<LU).

Fig. 9 Comparison of Xen-virtualization overhead (physical vs. virtual (8 VCPUs)) according to benchmark characteristics: (a) Kernel (version 2.6.18) compilation time with varying number of processes. (b) Normalized file copy time onto the same local disk volume with varying file size. (c) Network microbenchmarks with varying message size (roundtrip latency and oneway bandwidth).

head (time needed for kernel compilation), disk overhead (time needed for file copy), and network overhead (latency and bandwidth) measured in VMs as a ratio to that measured in physical nodes. The result here clearly shows that the overhead of VMs mainly comes from the disk and network I/O, supporting the message from Fig. 8, i.e., the workload with a higher communication ratio causes larger degradation in performance. The reason why the VM’s I/O causes the major overhead, can be found from the I/O subsystem of Xen Hypervisor that is based on the split driver model [45]. Here, only one privileged VM (Domain-0) is allowed to access physical I/O resources while all other VMs (Domain-U) must forward the request to Domain-0 whenever the access to resources is needed. When a parallel job involves a larger communication ratio and a larger number of CPUs, the wall-time cannot but increase since the Domain-0 VM not only receives more requests for network I/O, but also needs to process all the requests called from Domain-U VMs. Since it is well known that the VMM-bypass I/O techniques such as PCI-passthrough and single-root I/O virtualization (SR-IOV) significantly reduce the I/O overhead [46], we plan to work to improve the resource virtualization framework in near future.

Overall performance of the service: To discuss the quality of the trial service in an overall viewpoint, we present a couple of statistical datasets in Fig. 10 that have been obtained during the service period. First, the monthly distribution of all the 4,220 simulation jobs (top of Fig. 10(a)) shows that the service utilization has been quite consistent with no remarkable fluctuations in the usage frequency, indicating the platform has been under a consistent stress test during the trial service. Roughly 19.2% of all the processed jobs (809 jobs) ran on parallel while the other 80.8% (3,411 jobs) were serial as shown in bottom of Fig. 10(a). 86.7% of the submitted requests (3,658 jobs) have been completed successfully, while the others have been either cancelled (7.2%, 301 jobs) by users or failed (6.2%, 261 jobs) due to users’ fault in setting input parameters. Since no failed jobs have been observed due to technical issues of the platform so far, Fig. 10(c) supports that the operation of the platform has been quite stable.

Figure 10(d) shows the flow of times taken to com-
complete jobs successfully upon the start (top), and times that submitted jobs had to wait for until they start to run (bottom). The successful 3,658 successful jobs have been running for ~5,890 seconds (top) and have been waiting for ~0.6 seconds (bottom) on average. The extremely small average waiting time here indicates that the service has been quite prompt in processing user requests for simulations.

4. Conclusions

In this paperwork, we have discussed the current status of the EDISON science gateway project ongoing in Republic of Korea under governmental support. Along with an overview of the hierarchy, we provide in-depth explanations for technical details of the EDISON platform focusing on the three key features, i.e., native web-interface, Science Appstore, and virtualized resource/job management framework, which handle job requests overhead-efficiently and support domain-neutral TCAD software with great expandability.

The performance of resource virtualization has been demonstrated by benchmarking the performance against physical nodes with NAS parallel workloads, and we observe that VMs exhibit degradation in performance if they are coupled to workloads of a higher communication ratio. Based on the experimental result on the virtualization overhead, we confirm that the overhead mainly comes from the disk and network I/O, and suggest a strategy for improving the VM performance as a near-future working item. The stability and promptness of the EDISON platform in supporting simulations, have been also confirmed from the statistical dataset obtained from a 3-month trial service in the area of computational nanoelectronics (EDISON_Nanophysics).

In near future, we are going to start a full service of EDISON_Nanophysics (http://nano.edison.re.kr) and expand the service area to computational chemistry and computational fluid dynamics. Through the establishment of a virtuous system with convergence between research and education, we firmly believe that this work will lay the foundation for enhancing the national competitiveness in science and engineering.

Acknowledgements

This work has been carried out under the support from the EEducation-research Integration through Simulation On the Net (EDISON) project funded by the Ministry of Science, ICT and Future Planning, Republic of Korea (Grant Number: 2011-0020576). H. Ryu acknowledges the extensive use of TACHYON-II HPC cluster for testing and benchmarking TCAD software, and would like to dedicate this work to J.H. Sohn for her consistent encouragement and supports.

References

2003.
[34] Paraview Project. Available from http://www.paraview.org
[36] Linear Algebra PACKage (LAPACK), http://www.netlib.org/lapack
[38] ARnoldi PACKage (ARPACK). Available from http://www.caam.rice.edu/software/ARPACK

Hoon Ryu received the BSEE degree from Seoul National Univ., Republic of Korea in 2003, and the MSEEE and PhD degree from Stanford Univ., USA in 2005 and from Purdue Univ., USA in 2011, respectively. Currently he is a senior researcher at National Institute of Supercomputing and Networking, Korea Institute of Science and Technology Information. His research interests are nanoelectronics modeling and parallel computing with performance optimization.

Jung-Lok Yu received his Ph.D. degree from Korea Advanced Institute of Science and Technology (KAIST), Republic of Korea in 2007. He worked as a senior engineer in Samsung Electronics from 2007 to 2010. He is currently a senior researcher at National Institute of Supercomputing and Networking, Korea Institute of Science and Technology Information. His research interests include parallel processing, cluster computing, and cloud computing.

Doseok Jin is a senior researcher at National Institute of Supercomputing and Networking, Korea Institute of Science and Technology Information. He received his M.S. degree in Computer Science from Chonbuk National Univ., Republic of Korea and his Ph.D. degree in Computer Science from Paichai Univ., Republic of Korea in 2001 and 2011 respectively. His research interests cover information retrieval systems, cloud storages and distributed file systems.

Jun-Hyung Lee received his B.S. and M.S. degrees in Computer Engineering from Kyung Hee University, Republic of Korea in 2010 and 2012, respectively. Currently, he works as a researcher at National Institute of Supercomputing and Networking, Korea Institute of Science and Technology Information. His research interests are cloud computing and web services.

Dukyun Nam received the BS degree in computer science and engineering from Pohang Univ. of Science and Technology, Republic of Korea in 1999, and the MS and PhD degrees in engineering from KAIST ICC, Republic of Korea in 2001 and 2006, respectively. He is currently a senior researcher at National Institute of Supercomputing and Networking, Korea Institute of Science and Technology Information. His research interests are distributed systems, parallel computing, and cyber environment.

Jongsuk Lee is a principal researcher at National Institute of Supercomputing and Networking, Korea Institute of Science and Technology Information and an adjunct faculty at Univ. of Science and Technology of Korea. Her research interests are smart learning, parallel/distributed computing, and big data handling. She received her PhD degree in computer science from Univ. of Canterbury, New Zealand.
Kumwon Cho is a director of Supercomputing R/D Center, National Institute of Supercomputing and Networking, Korea Institute of Science and Technology Information. He received his PhD degree in Mechanical (Aerospace) Engineering from KAIST, Republic of Korea.

Hee-Jung Byun received the Ph.D. degrees from Korea Advanced Institute of Science and Technology (KAIST), Republic of Korea in 2005. She was a senior engineer in Samsung Electronics, Ltd. from 2007 to 2010. She is currently an assistant professor with the Department of Information and Communications, Suwon University, Republic of Korea. Her research interests include network modeling, controller design, and performance analysis.

Okhwan Byeon received the B.S. and Ph.D. degrees from Hankuk Aviation University and Kyunghee University, Republic of Korea in 1979 and 1993, respectively, all in network management and security areas. During 1979-1999, he was with High-Performance Networking Department at KIST and Supercomputing Application Department at ETRI. From Nov. 1999, he joined KISTI, where he currently works as a principal researcher. He is also a former-director of National e-Science Division in KISTI. He is a life member of Korean Academy Societies such as KIPS, KICS and KIISC since 1991. His research interests are e-Science platform and its Services, SDN with OpenFlow.