ELSEVIER

Advances in Engineering Software 35 (2004) 249–259

ADVANCES IN ENGINEERING SOFTWARE

www.elsevier.com/locate/advengsoft

A web-based platform for computer simulation of seismic ground response

Zhaohui Yang, Jinchi Lu, Ahmed Elgamal*

Department of Structural Engineering, University of California at San Diego, La Jolla, CA 92093 0085, USA

Received 2 June 2003; accepted 4 March 2004

Available online 13 April 2004

Abstract

The Internet provides an open environment for more efficient development and utilization of engineering software. This article presents a generic web-based platform for conducting model-based numerical simulations online. The platform distributes pre- and post-processing components to the user computer, and only retains core computational functions on the server machine. Design of this platform addresses Internet-specific issues such as supports for multiple users, integration of various programming languages or modules on both client and server sides, and the concerns of Internet traffic/security. As an implementation of this platform, a web site is developed for online execution of a solid–fluid fully coupled nonlinear Finite Element code, to conduct simulations of seismic ground response and liquefaction effects. At this web site, users can select the soil composition and input seismic excitation from built-in material/motion libraries, or define their own material properties and/or input motions. The output interface allows graphical rendering of simulation results, animations, and automated report generation. All software packages employed in this work are well tested and documented freeware, and can be easily adapted for execution of other computational codes.

Keywords: Internet computing; World Wide Web; Software engineering; Java; Graphical user interface; Liquefaction; Seismic response

1. Introduction

The Internet has introduced a robust real time mechanism for communication and interaction. In the near future, development and deployment of the Next Generation Internet (http://www.ngi.gov/) will lay down the infrastructure for a worldwide communication network that is even faster and broader than most of today's local networks. Ultimately, computers, data storage systems, local networks, and other resources will be connected by the Internet as a single massive system (http://www.npaci.edu/teragrid/).

These advances in communication infrastructure/hardware have allowed computer applications to be divided into components and distributed effectively over the Internet [1,2]. Today, most web-based computer programs leave user-interfacing tasks on the user (client) computers, and retain core processing functions on the server machine [3–6]. The interaction between users and central processing machines is accomplished through the client–server communication protocol of the World Wide Web (WWW), a standard software interface overlying the Internet infrastructure. As the Next Generation Internet matures, even the core computations can be further parallelized and distributed over the Internet [6,7]. Thus, technologies developed for high performance computing within a local network or inside a supercomputer can be extended to the Internet environment. It is also anticipated that by then, Internet-enabled real time computation and visualization will become commonplace so that user intervention (input interface), computation, and visualization (output interface) all occur concurrently with negligible time delay.

Although relatively scarce, applications of Internet-enabled techniques to Finite Element (FE) computer simulations are fairly straightforward [8]. A typical FE simulation involves three main phases: (1) input phase: defining the FE model, (2) computation phase: executing the FE code, and (3) output phase: viewing/analyzing computational results and writing a report. In the WWW environment, it is logical to distribute the phases 1 and 3 (i.e. user interfaces) to the client-side as shown in Fig. 1.

This article describes the framework of a generic web-based computational platform for conducting model-based simulations on line, taking advantages of well-established web programming tools. The design of this platform takes

* Corresponding author. Tel.: +1-858-822-1075; fax: +1-858-822-2260.
E-mail address: elgamal@uscd.edu (A. Elgamal).

0965-9978/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.advengsoft.2004.03.002

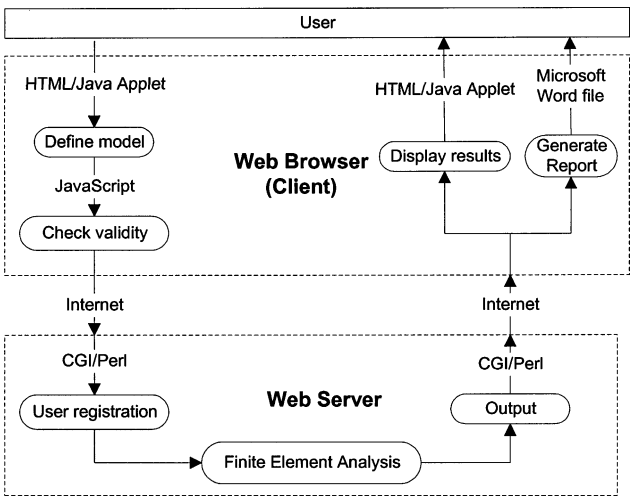

Fig. 1. Schematic of web-based simulation platform.

into account factors specific to the web environment, such as supports for multiple users, integration of various programming languages or modules on both client and server sides, and the concerns of Internet traffic/security. As a specific implementation, the web-based Cyclic1D simulation platform (http://cyclic.ucsd.edu), is described in some detail. Cyclic1D is a FE program for conducting computer simulations of nonlinear seismic ground response including liquefaction effects. Such simulations provide critical information for earthquake-resistant structural design in seismically active areas. A brief account of the theoretical background of Cyclic1D is also included in the Appendix A.

All software packages employed in the Cyclic1D simulation platform are well tested and documented freeware. These packages and corresponding download locations are listed in Table 1. A newly developed Windows version of Cyclic1D is also available at the same web site for downloading.

2. Web-based platform for Finite Element simulations

The design and implementation details of the three main phases shown in Fig. 1, i.e. input interface, computation, and output interface, will be the focus of this article. To implement a web-based application, a web server is first installed on the host machine. The web server is responsible for communications with client applications such as a web browser, and for the computational and communication tasks on the server. Various web servers are available for selection, such as the most popular Apache HTTP server (http://www.apache.org/). Besides the web server, a number of web-enabled programs need to be developed in order to carry out the operations shown in Fig. 1, to be described below.

Further extensions to this web-based platform (Fig. 1) can be made to allow for additional components to be connected and to interact. For instance [6], the web server may link to a database server to store user information and input/output data. Moreover, the core FE program may acquire element/material modules or analysis algorithms from other server machines during run time. Thus, the structure in Fig. 1 should be regarded as a basis for more general applications, rather than a fixed framework.

2.1. Input interface

Of the three main phases in a typical FE simulation (Fig. 1), phase 1 is responsible for collecting user input data to define the computational model. The input interface is almost exclusively implemented as web pages using HTML language, which is accessed by the user via a web browser (e.g. Internet Explorer or Netscape). These web pages allow the user to define a model by making selections, entering numbers or text, or uploading his/her own input files. Upon completion of the input phase, the user clicks on a 'Submit' button, and the web browser will: (1) submit all user data to the web server and (2) invoke the first operation on the server side. The first task is automatically accomplished through Internet data transmission protocols (e.g. TCP/IP). The second task is fulfilled by associating with the Submit button an appropriate sever side program.

Although, the implementation appears to be straightforward, it is important to validate the user data before submitting it to the server. Leaving model validation on the client-side reduces unnecessary Internet traffic, as well as processing load on the server. Data validation can be performed by client-side Java Script embedded in the web pages and invoked either before or after clicking the Submit button. For instance, Fig. 2 shows that upon clicking the Submit button, a client-side Java Script function dataCheck( ) is called upon to perform model validation. The validated model is submitted, and a sever side program

Table 1
Employed freeware packages and web-based programming languages

Usage	Package/language	Download/documentation site
X–Y plotter/animation	PtPlot	http://ptolemy.eecs.berkeley.edu
GIF generator	GNUPLOT	http://gnuplot.info
web server	Xitami	http://xitami.com
Server side programming languages	CGI Perl	http://hoohoo.ncsa.uiuc.edu/cgi; http://perl.com
Client-side programming languages	HTML Java Script	http://www.w3.org/markup; http://java.sun.com

```
<html>
<head>

// JavaScript performing client-side data validation
<script language="JavaScript">
function dataCheck()
{
  if (profileHeight<0) return false;   //something is wrong
  ...
  return true;                         // model is valid
}
</script>
</head>

// Upon clicking the "Submit" button, the above-defined
// JavaScript function dataCheck() is called.
// If the model is valid, it will be submitted, and
// the server-side program "user_register.pl" will be invoked.
<form action="user_register.pl" method="post" onsubmit="return dataCheck()">
...

// Create the "Submit" button
<input type="submit" value="Data check and Submit">
...
</form>
</html>
```

Fig. 2. Client-side data validation, model submission, and invocation of sever side routine for user identification (written in HTML).

'user_register.pl' is invoked to perform user registration (see below).

2.2. Server side operations

There are mainly three tasks on the server side (Fig. 1): (1) user registration; (2) simulation; and (3) generation of output interface. For smaller applications, a single server machine may be sufficient to handle all three tasks. For larger applications, these tasks can be distributed to a cluster of server machines [6].

2.2.1. Server side programming tools

The sever side tasks cannot always be performed by a single program or programs written in the same language. For instance, a FE computational code is typically a compiled binary executable file written in Fortran or C++, whereas it is more convenient to handle user registration and generation of output interfaces using a scripting language. All web servers comply with the Common Gateway Interface (CGI), a standard sever side programming environment allowing programs written in virtually any language to be executed [9].

As to the choice of a scripting language, there are a number of candidates (e.g. Perl, Visual Basic, Java Script, Tcl, etc.). By far, Perl is the leading programming language for web-based applications ([10], Table 1). In fact, there are many user-developed Perl libraries/modules existing in the Internet that can be directly adopted or customized for specific applications. Consequently, Perl was selected as the main programming tool for the Cyclic1D web site to be discussed below.

2.2.2. User registration

User registration is necessary for maintaining a multiple-user environment, and typically should be the first operation invoked by a user request. In order for multiple users to conduct numerical simulations simultaneously, separate storage space on the server machine is needed for each registered user to store model data. When a user logs in for the first time, dedicated disk space is assigned to that user. All subsequent simulation requests from that user will be directed to this storage allocation. Therefore, an identity is necessary to map each incoming user request to a specific disk location.

Generally, there are two methods for tracking user identities, either explicitly or implicitly. Explicit user registration requires the user to provide a name and/or a password, which is used later to access the corresponding storage allocation. An advantage of this method is that the user can log on from anywhere on the Internet. On the other hand, implicit user registration does not require a user name. Instead, a user is identified through certain hidden mechanism (e.g. using the IP address of the client computer or storing a cookie on the client computer). Thus, the same user will be treated as a different person each time he/she logs on from a different computer. However, this method is simpler and is suitable for applications where only short-term disk usage is expected.

2.2.3. Simulation

Once the disk allocation is made, the user input data will be forwarded there for further processing. As shown in Fig. 3, a driver routine prepares input data files based on the user data, and then launches the core FE code to conduct the simulation. After the simulation is completed, a number of additional programs may be called upon for

```
# Open CYCLIC input file
open(OUT, ">CYCLIC.in") || die;

# Write model parameter values into the file
print OUT $PROFILEHEIGHT, $NUMELEMENTS, $WATERDEPTH;
print OUT $INCLINATION, $BEDROCK, $AM, $AK;
...

# Close input file
close(OUT);

# Execute core CYCLIC code
system "CYCLIC.exe < CYCLIC.in";

# Compute Fourier Transform and response spectrum
system "FFT.exe >FFT.out";
system "Respect.exe >respect.out";

# Generate dynamic web pages for output interface
print <<END_HTML;
<html>
<head>
<title>Results of CYCLIC1D Simulation: Summary</title>
</head>

<body background="/pics/background.gif">
...
</body>
</html>
END_HTML;
```

Fig. 3. Sever side driver routine for creating input files, launching FE program, and writing output interface (written in Perl).

post-processing. Finally, the driver routine writes an output interface to the user. Although the driver routine as shown (Fig. 3) was written in Perl, other languages may serve the purpose as well.

While a simulation is running, it is important to notify the user of its status. One approach is to generate a web page that periodically updates itself. This can be done conveniently through either the client-pull or server-push mechanism supported by most web browsers. For example, in the case of a client-pull, a 'Refresh' derivative is inserted in the header of the web page (Fig. 4). This web page will then periodically reload itself in a specified number of seconds. However, for simulations that take more than several minutes, it becomes impractical to keep the user waiting on line. In such cases, a mechanism can be implemented to notify the user once the simulation is completed (e.g. by e-mail).

Note that the periodical user updating mechanisms may potentially be extensible to real time applications. In this case, the server will continually deliver up-to-the-second simulation results to the client as a smooth data stream, so that the user can visualize the results over the Internet while the computation is still running. In the near future, with significant speedup in computation and data transmission, such a scenario can become commonplace.

2.3. Output interface

The output interface is generated as dynamic web pages (Fig. 3) and sent back to the user through the Internet. The main purpose of this interface is for the user to conveniently visualize and manipulate the simulation results. For example, the user may be allowed to download the results as data files, figures, or reports. A variety of web programming tools is available for animation and interactive visualization such as Java Applet, animated GIF, etc.

Simulations often generate large output data sets, and it becomes time consuming to transmit all this data to the user. Moreover, in most cases the user might be interested only in a selected portion of the results. Therefore, the output interface should allow the user to decide on what information to extract. One way to achieve this is by automatic generation of a customized report that collects all desired information about the simulation. The report may be written in any suitable format such as Microsoft Word.

```
# File name: track.pl
...
# Calculate percentage of the simulation completed.
$percent = $time_processed*100/$time_total;

print <<END_HTML;
<html>
// client-pull by inserting a "refresh" derivative.
// In this case, the web page will reload "track.pl"
// in every 10 seconds.
<meta http-equiv="Refresh" content="10; URL=track.pl">

<body background="/pics/background.gif">
The simulation has completed $percent percent.
...
</body>
</html>
END_HTML;
```

Fig. 4. Client-pull mechanism to periodically update the user on the simulation status (written in Perl).

As an implementation of the web-based simulation platform outlined above, we developed a web site (http://cyclic.ucsd.edu) for online operation of Cyclic1D, a nonlinear FE program for numerical simulation of seismic ground response including liquefaction effects (Fig. 5). The implementation details of this web site are presented below.

3. Implementation: Cyclic1d web site

Experience from past strong earthquakes worldwide has distinguished soil liquefaction as one of the main causes of structural damage [11–14]. In recent years, a number of computer programs has been developed for assessing earthquake-induced nonlinear ground response including liquefaction effects (e.g. DYSAC2 [15], DYNAFLOW [16], SUMDES2D [17], CYCLIC [18]). However, even with great advances in computational capabilities, usage of these programs is still relatively limited. One main reason is that the underlying soil constitutive models usually require a large number of input parameters (10–20 typically for each soil material type), and a lengthy calibration process. In addition, analysis of large amount of data generated from these simulations demands efficient tools. Consequently, a user-friendly interface for convenient pre- and post-processing is essential.

In view of the above need, the Cyclic1D web site (Fig. 5) was developed aiming to greatly simplify user interfaces, without undue compromise on modeling flexibility. At the input interface, soil materials are classified into 15 categories, each with a set of pre-defined material constants. Thus, the typical user is relieved from an otherwise much-involved calibration process. Moreover, the user may define an input base excitation either from a built-in library or by uploading his/her own file. To assist the user in processing the results, the output interface features online graphical data rendering, animation, and automated report generation. A full accounting of the implemented user interfaces is presented as follows.

3.1. Input interface

The input interface is implemented as an interactive web page using HTML language. The user defines and submits a FE model using a web browser such as Internet Explorer or Netscape. A FE model is defined by specifying: (1) the soil profile of interest; (2) material composition of the profile; (3) Rayleigh viscous damping coefficients; and (4) base seismic excitation.

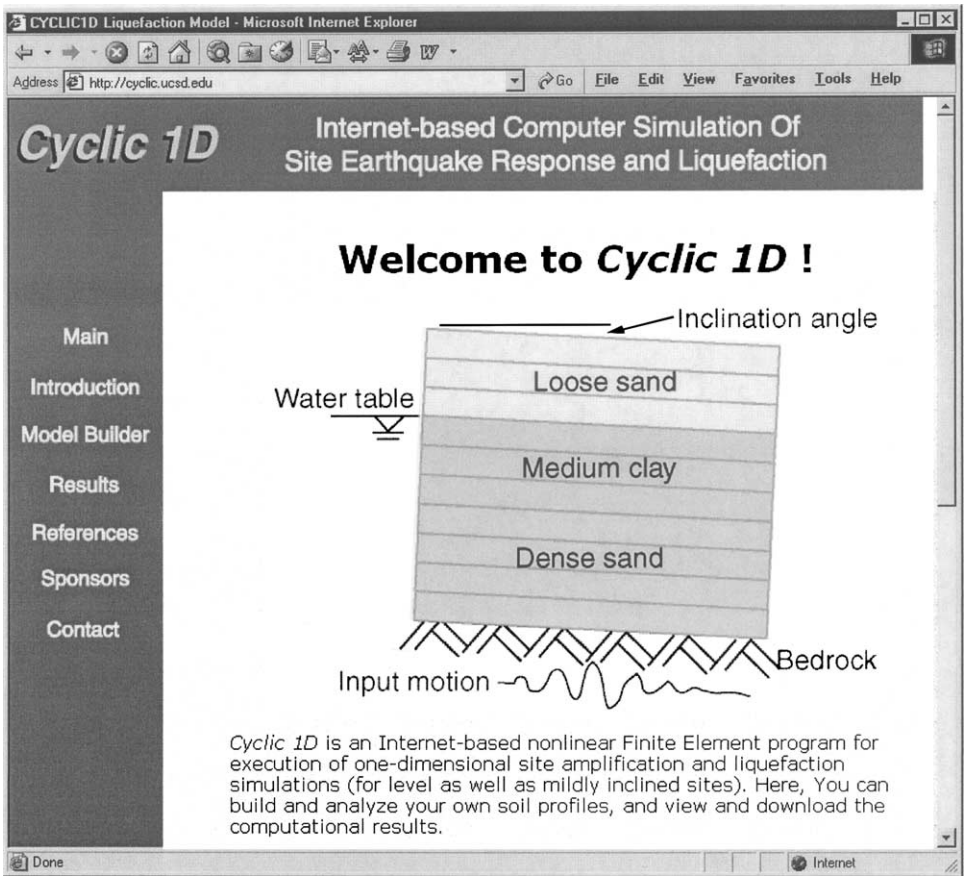

Fig. 5. Cyclic1D web site (http://cyclic.ucsd.edu) for on line simulation of one-dimensional seismic ground response including liquefaction effects.

3.1.1. Model profile

The following user input options are available in the model profile section (Fig. 6):

(1) Soil profile height (any value from 5 up to 100 m).
(2) Number of elements (10–100 elements): more elements allow for simulations of higher spatial resolution.
(3) Depth of water table: soil above the water table is modeled as dry material and soil below as saturated. A 0.0 m depth represents water table at ground surface.
(4) Inclination of soil profile (any value from 0.0 to 10.0°): for mild infinite-slope simulations, with a 0° inclination representing level ground.
(5) Bedrock property: bedrock material below the soil profile (Fig. 6) can be specified as 'Rigid', 'Hard

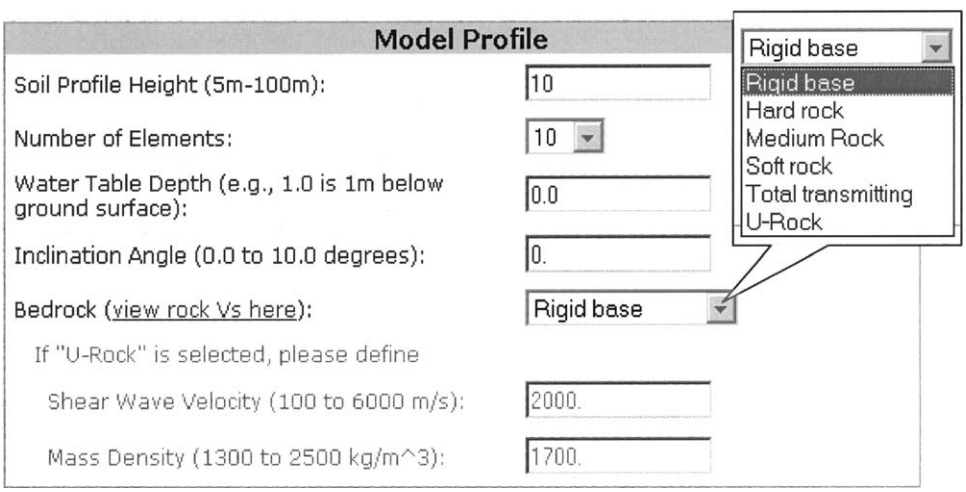

Fig. 6. User dialog window for defining model profile.

Rock', 'Medium Rock', 'Soft Rock', 'Totally Transmitting', or 'U-Rock' (user-defined bedrock type). If 'Rigid', the input motion (see below) is treated as a total motion. Otherwise, the input motion is handled as a rock outcropping motion (Schnabel et al. 1972). For 'U-Rock', the user also needs to specify shear wave velocity and mass density of the bedrock (Fig. 6).

3.1.2. Soil properties

A complete definition of each soil type requires about 15 modeling constants in the core FE code. Considering the large number of constants involved, we have pre-defined model parameters for typical soil types in the Cyclic1D input interface (Fig. 7). Definition of these constants was based partially on an intensive calibration phase (Appendix A), and partially on data from the available literature.

The pre-defined materials fall into two main categories: cohesionless and cohesive. For cohesionless materials, it is known that relative density and permeability are among the most influential parameters controlling nonlinear stress–strain behavior and liquefaction response [19]. Therefore, we defined four cohesionless soil types covering a wide range of relative densities: loose (representative of relative densities between 15 and 35%), medium (35–65%), medium-dense (65–85%), and dense (85–100%). Furthermore, each of the four types is associated with three different permeability coefficients (representative of silt, sand, and gravel, respectively), resulting in a total of 12 materials. For cohesive materials, there are three types based on shear strength: soft, medium, and stiff clay. In addition, the user can define up to five clay (or rock) materials (U-clay/rock) by specifying mass density, shear strength and shear wave velocity. Different materials may be assigned to each individual element.

3.1.3. Additional viscous damping

In Cyclic1D, damping is mostly generated from soil nonlinear hysteretic response. Additional Rayleigh-type viscous damping may be assigned either by directly specifying two Rayleigh damping coefficients, or by way of specifying two damping ratios at two different frequencies [20]. After these coefficients are defined, the corresponding damping ratio curve is portrayed as a function of frequency in a dialog window (Fig. 8). This useful visual feature allows the user to define interactively the desired dependence of damping on frequency.

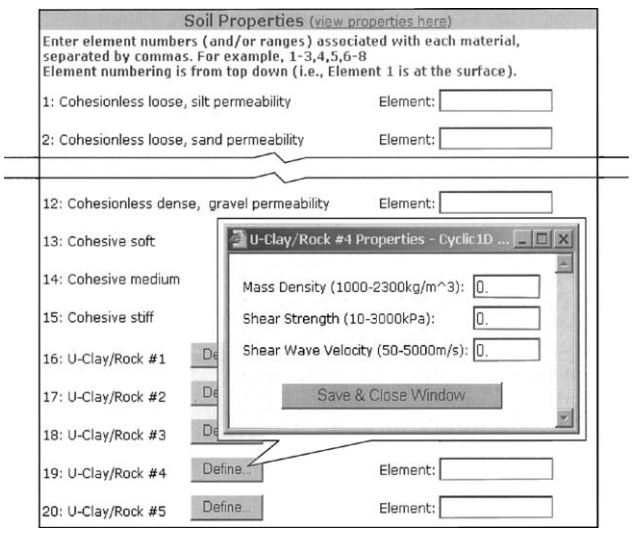

Fig. 7. User dialog window for defining soil material properties.

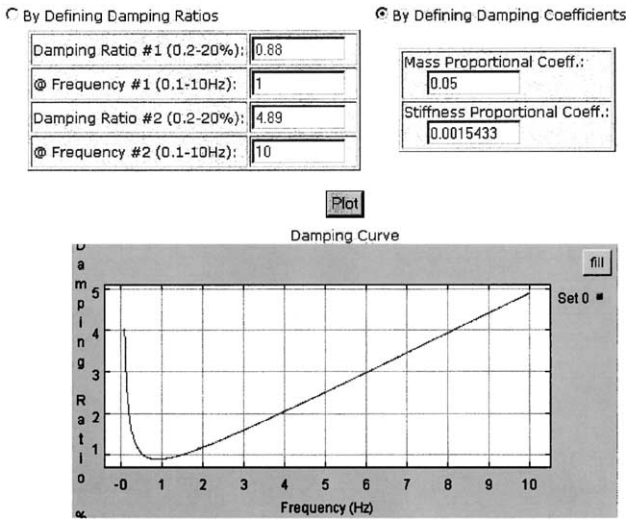

Fig. 8. User dialog window for defining Rayleigh damping coefficients and viewing damping ratio curve as a function of frequency.

3.1.4. Input motion

Base seismic excitation can be defined by either of the following two methods (Fig. 9a):

(1) Via a built-in input motion library. This library includes near-fault soil surface motions as well as long-duration rock outcrop motions recorded during past strong earthquakes worldwide, as described in detail at http://peer.berkeley.edu/research/motions/ (where these motions are available for downloading).
(2) 'U-Shake', a user-defined input motion. The user can upload an input motion data file from a local disk drive (Fig. 9b). This file will be screened by the web server to ensure valid formatting.

The amplitude of the input motion can be scaled by a factor ranging from 0.01 to 1.0. In addition, if '1g sinusoidal motion' is chosen, the user must specify excitation frequency and number of cycles.

3.1.5. Data validation

Once the user defines the model and clicks the Submit button, a Java Script routine is triggered on the user computer to validate this model (Fig. 2). Mainly, the routine checks: (1) if any input parameter value is out of its specified range, and (2) if any FE is associated with more than one material or no material at all. If the model is valid,

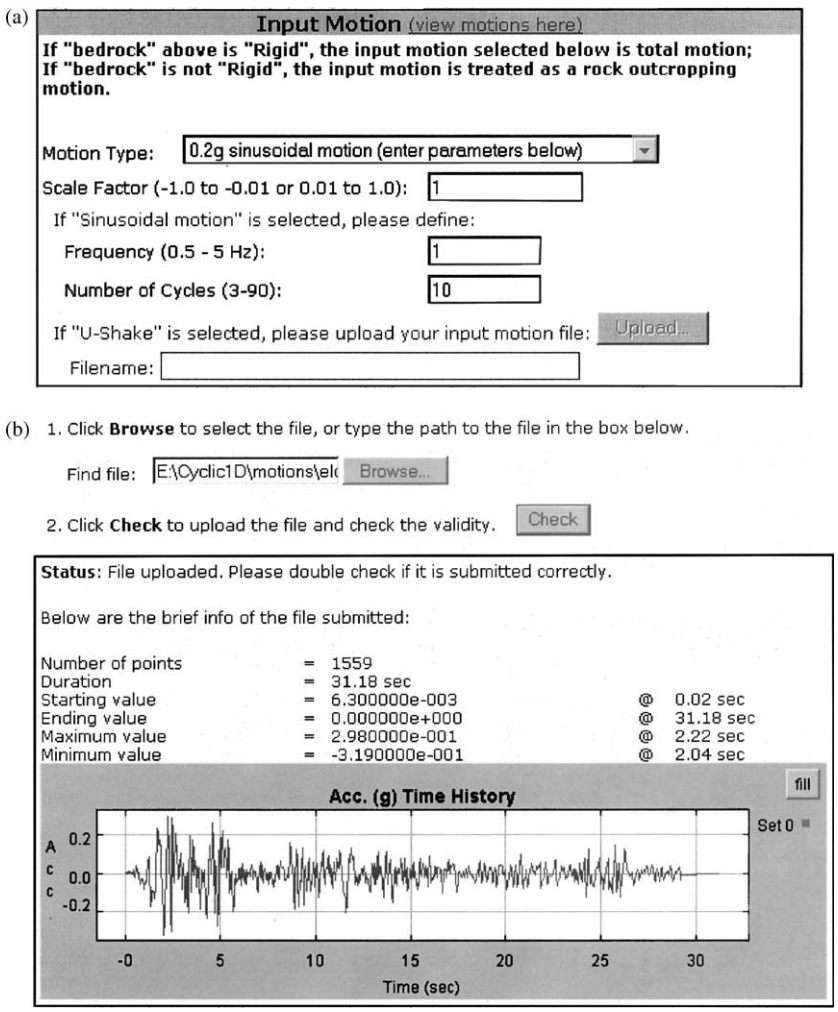

Fig. 9. (a) User dialog window for defining input motion. (b) User dialog window for defining U-Shake (user-defined input motion).

the input data are then submitted to the web server for simulation. Otherwise, the user is required to make corrections.

3.2. Simulation

On the server side, the Xitami HTTP server (Table 1), a robust portable Open Source package, was employed. Other popular web servers such as Apache or Microsoft IIS may be used as well.

User identification/registration is the first operation triggered by a simulation request. This is done in Cyclic1D by keeping track of the IP address of user machines. A sever side routine checks the sender's IP address against a list of registered users. If the user is already on the list, the simulation request will be directly forwarded to the corresponding disk space for further processing. Otherwise, the user will be added to the list and assigned a new allocation. Note that a user registration expires if this user has been inactive for a period of time (currently set to 1 day).

A driver routine (written in Perl) then prepares input data files based on the user data, and launches the core FE code to conduct the simulation (Fig. 3). During the simulation, the user is constantly updated on the status of the process, using the client-pull technique described above (Fig. 4). Once the simulation is completed (which usually takes only a few minutes), the driver routine calls a number of additional programs to process the results (e.g. calculating Fourier Transform and response spectrum, Fig. 3). Finally, the output interface is generated as dynamic web pages and sent back to the user (Fig. 3).

3.3. Output interface

Many users are interested in response time histories at a particular depth (e.g. ground surface). Such time histories include acceleration (and its response spectrum and Fourier spectrum), displacement, excess pore pressure, shear stress, and shear strain (Fig. 10). In Cyclic1D, the user can view all these histories for any desired depth in one window (Fig. 11). Moreover, the user can: (1) download any of these histories

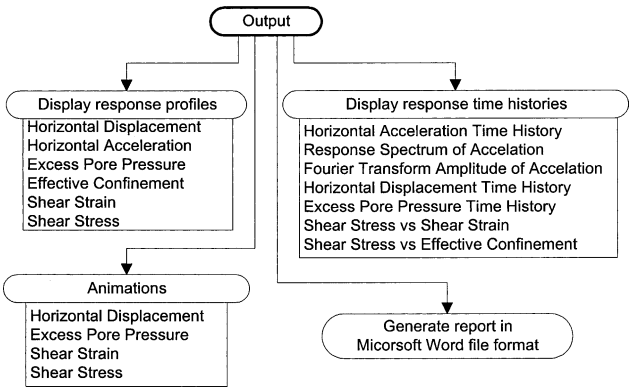

Fig. 10. Available Cyclic1D output interfaces.

as a figure (images in graphics interchange format or GIF) and/or a data file, and (2) select any of these histories to be included in a report (Section 3.4).

In addition to time histories of individual variables, the user can also view the maximum and final values of these variables along the model depth (i.e. response profile or response envelope, Fig. 12). These response profiles help the user appreciate overall performance of the model. Again, the user can include any of these response profiles in the report. All results can be downloaded as data (ASCII) files or images. Moreover, the user can view animations of horizontal displacement, excess pore pressure, shear strain, and shear stress responses along the entire model profile.

The following software packages were employed in implementing the Cyclic1D interfaces (Table 1): (1) interactive X–Y plots and animations are generated using Ptplot, an open source plotting tool written in Java language [21], and (2) all images (GIF files) are created using the freeware package GNUPLOT (http://gnuplot.info).

3.4. Report generator

Instead of keeping all model input/output data on the server machine or directly downloading this data to the client in terms of many separate files, a convenient option is to write a report that includes all desired information about the simulation. Using the report generator function in the Cyclic1D output interface, a customized report can be created in Microsoft Word or Rich Text Format (RTF) for downloading and further modification. While the word file format is most convenient for client computers using a Windows operating system, the RTF format is compatible with many other operating systems including Unix and Macintosh.

The report generator window allows for including any portion of the model input/output information described above. In the resulting report, input model parameters are listed in tables, whereas the simulation results are presented as GIF images. The report generator was implemented using the Perl Win32::OLE module (http://aspn.activestate.com), which allows a Perl program to create, access, and modify

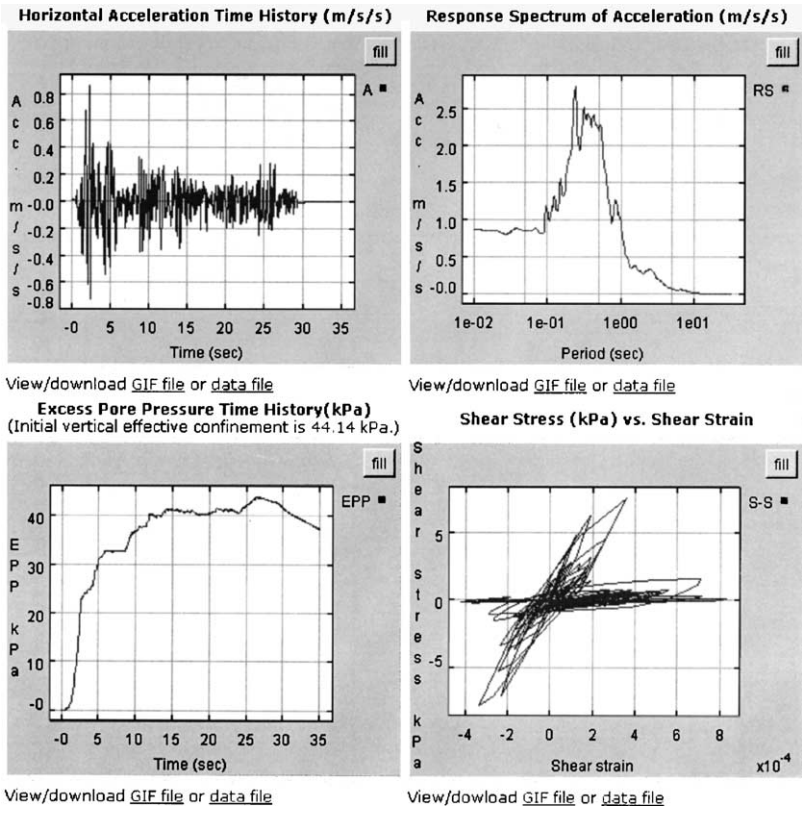

Fig. 11. Sample graphical output for response time histories.

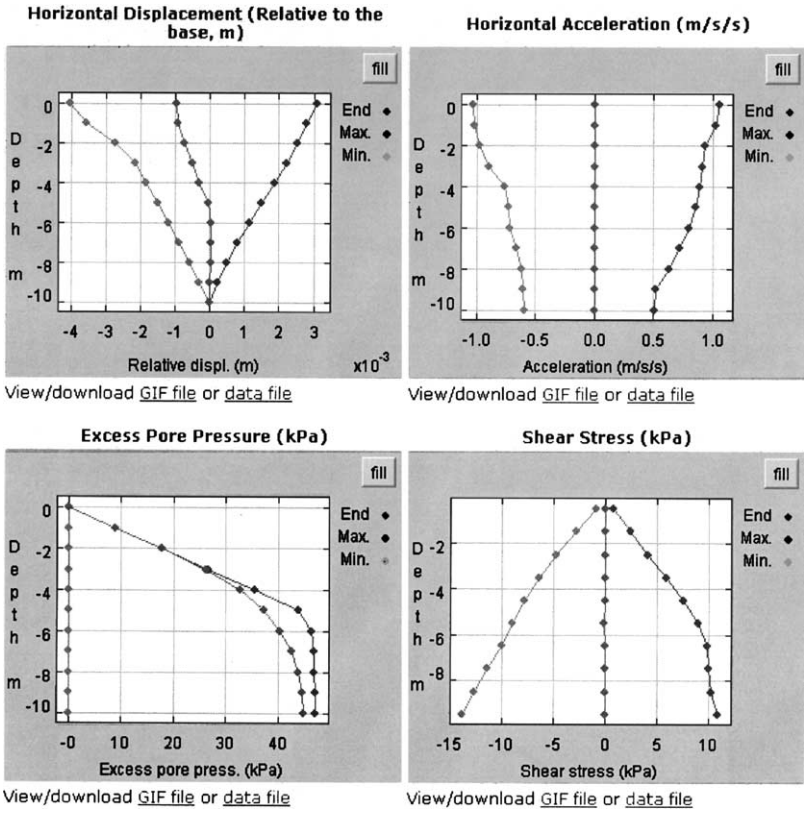

Fig. 12. Sample graphical output for response profiles.

many Win32 applications such as Word or Excel files. For web servers running on other operating systems that do not support word (e.g. Unix), alternative report file formats such as HTML may be considered.

4. Web-based versus standalone platforms

Compared to traditional standalone machine operations, current web-based computing may be slowed down by the Internet communication speed. Therefore, conducting a large number of interactive simulations in a short period of time may be somewhat impractical at present. It is anticipated that the high-speed Next Generation Internet will alleviate this problem [6]. In addition, due to the concern of Internet security, some user options (e.g. user-provided input motion) have to be implemented with caution. To this end, a number of security measures may be employed, such as:

(1) *User authentication*. User accounts can be protected by passwords. The IP address of the user machine can be checked as well, and login requests from unwelcome IP addresses can be rejected.
(2) *Virus scanning*. Any user uploaded file is first scanned by an anti-virus program before further processing.

In spite of these minor shortcomings, the advantages of web-based applications are overwhelming. From the user end:

(1) Web-based applications can be accessed anytime, anywhere worldwide from any platform (Windows, Unix, Linux, etc.).
(2) Web-based platforms can host many users and execute many user requests at the same time.
(3) The application programs are installed on the server machine so that users are spared the inconvenience associated with maintenance/upgrade of the software.
(4) The computed results can be achieved on the server and thereafter accessed from any other machine at any time.

From the developer end:

(1) Web-based platforms are extensible and scalable. Different software and hardware components may be easily connected to the central server via the Internet.
(2) Web-based platforms provide an open environment for efficient collaboration among researchers and developers, a significant advantage over the traditional software development paradigms.
(3) Performance of the software (as well as the hardware) can be easily monitored so as to improve its quality promptly.

(4) Copyright and other legal issues that are often troublesome for standalone programs are no longer an issue in the web-based environment (no copies of the code are distributed).

In view of the clear edge afforded by web-based computing, effort is currently underway to extend the Cyclic1D web site to allow for 2D and 3D simulations. To maintain high efficiency, large-scale 2D/3D simulations will be performed using appropriate parallel computing algorithms (and/or grid computing applications).

5. Summary and conclusions

A web-based platform was developed for conducting online model-based computer simulations. Design of this platform takes into account factors specific to the Internet environment, such as management of multiple users, integration of various programming languages or modules on both client and server sides, and the concerns of Internet traffic/security. An implementation of this platform, the Cyclic1D web site (http://cyclic.ucsd.edu), was presented that allows remote access to a solid–fluid fully coupled FE program for conducting nonlinear ground response and liquefaction simulations. The developed user interfaces provide libraries of pre-defined material properties and input motions, tools for viewing computational results, and automated report generation capabilities. The Cyclic1D web site has been used by many students and researchers worldwide. Effort is currently underway to extend this web site to allow for large-scale 2D/3D simulations, using appropriate parallel/grid computing algorithms.

The presented framework has demonstrated use of the Internet and the WWW as a viable and convenient user interface for computer simulations. This platform can be easily extended for implementation of other similar applications. Such web-based platforms allow users to access engineering software remotely from anywhere worldwide. Moreover, these platforms provide an open environment for efficient collaboration in developing large-scale software. In conclusion, web-based applications will continue to grow and become an important medium for civil and infrastructural engineering simulations.

Acknowledgements

The reported research was supported by the Pacific Earthquake Engineering Research (PEER) Center, under the National Science Foundation (Award Number EEC-9701568), and by the National Science Foundation (Grant No. CMS0084616). This support is most appreciated. Discussions with Professor Kincho Law and Dr Jun Peng of Stanford University are fruitful towards the development of the web-based platform.

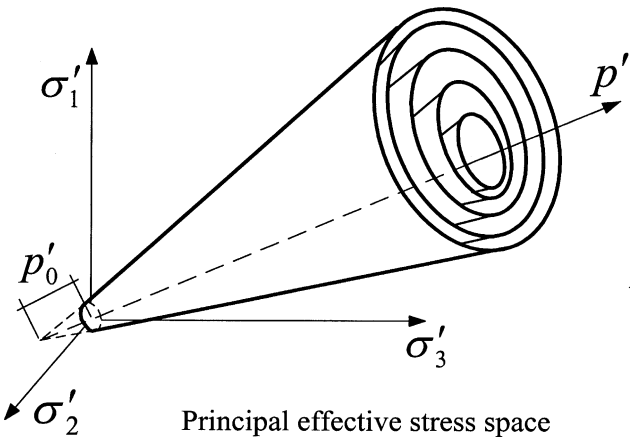

Fig. A1. Multi-surface plasticity model employed in CYCLIC [23,26].

Appendix A. The finite element code

Cyclic1D is a 1D version of the 2D FE code CYCLIC for simulation of seismic response of soil systems including liquefaction scenarios [18,22,23]. CYCLIC employs a two-phase (fluid and solid) fully coupled FE formulation [24], based on the Biot theory [25] for fluid-saturated porous media. In CYCLIC, the soil stress–strain behavior is governed by a new constitutive model [22,26] within the general framework of multi-surface plasticity (Fig. A1). In the new model, emphasis is placed on more accuracy in reproducing: (1) the salient cyclic stress–strain characteristics associated with shear-volume coupling (dilatancy)

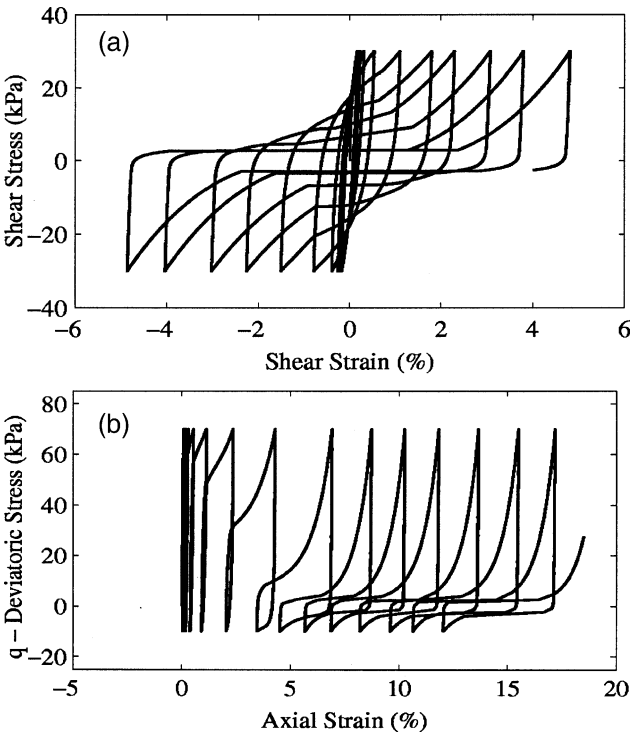

Fig. A2. CYCLIC simulations of (a) stress-controlled cyclic simple shear test, and (b) stress-controlled cyclic triaxial test with static stress bias [26].

effects exhibited by cohesionless soils when subjected to earthquake excitation, and (2) the permanent shear deformations accumulated during the soil liquefaction phase (Fig. A2). Reliable accounting of the magnitude of these deformations is of paramount importance for earthquake-resistant analysis and design.

Calibration and validation has always been an integral part of CYCLIC development. Experimental programs conducted on the Rensselaer Centrifuge (http://www.rpi.edu/~dobryr/centrifuge.html) have been a major source of calibration over the years (e.g. [18,23,27]). Laboratory soil sample data were also used [22,28,29]. In addition, actual earthquake response data recorded during past strong earthquakes worldwide (Japan, Taiwan, and USA) are being continuously used for calibration [29].

References

[1] Smith BL, Scherer WT. Developing complex integrated computer applications and systems. J Comput Civil Engng, ASCE 1999;13(4):238–45.

[2] Hopkins J. Component primer. Commun ACM 2000;43(10):27–30.

[3] Peng J, McKenna F, Fenves GL, Law KH. An open collaborative model for development of finite element program. Proceedings of Eighth International Conference on Computing in Building and Civil Engineering (ICCCBE-VIII), Palo Alto, CA; 2000. p. 1309–16.

[4] Peng J, Liu D, Law KH. An engineering data access system for a finite element program. Adv Engng Software 2003;34(3):163–81.

[5] Peng J, Law KH. Framework for collaborative structural analysis software development. Structural Congress and Exposition, ASCE, Philadelphia, PA; 2000.

[6] Peng J, Law KH. A prototype software framework for Internet-enabled collaborative development of a structural analysis program. Engng Comput 2002;18(1):38–49.

[7] Han CS, Kunz JC, Law KH. Building design services in a distributed architecture. J Comput Civil Engng, ASCE 1999;13(1):12–22.

[8] Nuggehally M, Liu YJ, Chaudhari SB, Thampi P. An Internet-based computing platform for the boundary element method. Adv Engng Software 2003;34(5):261–9.

[9] Gundavaram S. CGI programming on the World Wide Web. Sebastopol: O'Reilly; 1996.

[10] Wall L, Christiansen T, Schwartz R. Programming Perl, 2nd ed. Sebastopol: O'Reilly; 1996.

[11] Seed RB, Dickenson SE, Riemer MF, Bray JD, Sitar N, Mitchell JK, Idriss IM, Kayen RE, Kropp A, Harder LF Jr, Power MS. Preliminary report on the principal geotechnical aspects of the October 17, 1989, Loma Prieta earthquake. Report No. UCB/EERC-90/05,Earthquake Engineering Research Center. Berkeley: University of California; 1990.

[12] Japanese Geotechnical Society, Special issue on geotechnical aspects of the January 17, 1995 Hyogoken-Nanbu earthquake. Soils Found 1996;36(1):1–359.

[13] Japanese Geotechnical Society, Special issue on geotechnical aspects of the January 17, 1995 Hyogoken-Nanbu earthquake, No.2. Soils Foundations 1998;38(2):1–216.

[14] Ansal A, Bardet JP, Barka A, Baturay MB, Berilgen M, Bray J, Cetin O, Cluff L, Durgunoglu T, Erten D, Erdik M, Idriss IM, Karadayilar T, Kaya A, Lettis W, Olgun G, Paige W, Rathje E, Roblee C, Stewart J, Ural D. Initial geotechnical observations of the November 12,1999, Düzce earthquake. A Report of the Turkey–US Geotechnical Earthquake Engineering Reconnaissance Team; 1999.

[15] Muraleetharan KK, Mish KD, Yogachandran C, Arulanandan K. DYSAC2: dynamic soil analysis code for 2-dimensional problems. Davis: Computer Code, Department of Civil Engineering: University of California; 1988.

[16] Prevost JH. DYNAFLOW user's manual. Princeton, NJ: Department of Civil Engineering and Operations Research, Princeton University; 1998.

[17] Li XS, Ming HY, Cai ZY. Constitutive modeling of flow liquefaction and cyclic mobility. In: Arulanandan K, Anandarajah A, Li XS, editors. Computer simulation of earthquake effects. ASCE Geotechnical Special Publication, vol. 110.; 2000. p. 81–98.

[18] Elgamal A, Yang Z, Parra E. Computational modeling of cyclic mobility and post-liquefaction site response. Soil Dyn Earthquake Engng 2002;22(4):259–71.

[19] Kramer SL. Geotechnical earthquake engineering. Upper Saddle River: Prentice Hall; 1996.

[20] Chopra AK. Dynamics of structures, 2nd ed. Upper Saddle River: Prentice Hall; 2001.

[21] Lee EA, Davis J, Hylands C, Janneck J, Liu J, Liu X, Neuendorffer S, Sachs S, Stewart M, Vissers K, Whitaker P, Xiong Y. Overview of the Ptolemy project. Technical Memorandum UCB/ERL M01/11, Department of Electrical Engineering and Computer Science: University of California, Berkeley, See also http://ptolemy.eecs.berkeley.edu/; 2001.

[22] Yang Z. Numerical modeling of earthquake site response including dilation and liquefaction. PhD Dissertation, Dept. of Civil Engineering and Engineering Mechanics. New York: Columbia University; 2000.

[23] Yang Z, Elgamal A. Influence of permeability on liquefaction-induced shear deformation. J Engng Mech, ASCE 2002;128(7):720–9.

[24] Chan AHC. A unified finite element solution to static and dynamic problems in geomechanics. PhD Dissertation. UK: University of Wales, College of Swansea; 1988.

[25] Biot MA. The mechanics of deformation and acoustic propagation in porous media. J Appl Phys 1962;33(4):1482–98.

[26] Elgamal A, Yang Z, Parra E, Ragheb A. Modeling of cyclic mobility in saturated cohesionless soils. Int J Plast 2002;19(6):883–905.

[27] Dobry R, Taboada V, Liu L. Centrifuge modeling of liquefaction effects during earthquakes. In: Ishihara K, editor. Proceedings of First International Conference on Earthquake Geotechnical Engineering (IS-Tokyo), 3. Rotterdam: Balkema; 1995. p. 1291–324.

[28] Arulmoli K, Muraleetharan KK, Hossain MM, Fruth LS. VELACS: verification of liquefaction analyses by centrifuge studies, laboratory testing program, soil data report, Project No. 90-0562. Irvine, CA: The Earth Technology Corporation; 1992.

[29] Elgamal A, Lai T, Yang Z, He L. Dynamic soil properties, seismic downhole arrays and applications in practice. In: Prakash S, editor. State-of-the-art paper, Proceedings of Fourth International Conference on Recent Advances in Geotechnical Earthquake Engineering and Soil Dynamics, March 26–31, San Diego, CA. 2001.