Appendix A - User's Guide to Opensees

A.1 Introduction to Using OpenSees

The Open System for Earthquake Engineering Simulation (OpenSees) is a software framework created for the analysis of structural and geotechnical systems subjected to earthquakes. The code for the OpenSees software is open source, or freely available to anyone who wishes to develop the program. The open nature of the source code makes advances possible for a variety of advanced applications. OpenSees is a project sponsored by the Pacific Earthquake Engineering Research (PEER) Center for the purpose of continual research in performance-based earthquake engineering.

OpenSees was chosen as the analysis software package for the proposed research for a variety of reasons. Primarily, the analysis options and versatile code development of the software allow for direct application to increasingly complex and previously unstudied topics. The powerful analytical capabilities of the program combined with the unlimited versatility make it the ideal tool for nonlinear structural modeling. Also, because this software is developed in conjunction with PEER, it is relevant to a wide variety of earthquake related projects and will become increasingly useful for future research. The work required to learn this software package is justified due to the future applications to advanced projects through PEER.

OpenSees was created as a new paradigm for software analysis packages in which the source code is continually under development by the users. The continual development of the source code allows for the newest and most advanced structural analysis methods and objects to be added to the program as they are developed. The state of the code at any point in time represents the variety of applications to which the code has been adapted for analysis. Currently, OpenSees is capable of performing many types of finite element modeling, dynamic analyses, and nonlinear analysis of a wide range of structures and systems. OpenSees is currently being developed as the primary analysis package for pseudodynamic testing through PEER related research universities. As OpenSees is applied to a wider range of projects, it will continue to grow in its versatility, power, and usefulness.

A2 OpenSees Program Outline

OpenSees is comprised of a set of four modules to perform the necessary functions of an analysis software package. The Modelbuilder module is responsible for the creation of a finite element model. The Analysis module is where the specification of the analysis procedure takes place. The Domain keeps track of the selected quantities to be monitored during the analysis, and the Recorder object specifies the output of results. The relationship between these modules stems from the object-oriented programming standard from which OpenSees has been created. This format allows for a more versatile relationship between the objects within a program, and thus a more efficient and modifiable program.

The Domain object acts as the central operator while the program is being operated. This module is responsible for storing the objects created by the Modelbuilder object and for providing the Analysis and Recorder objects access to these objects during the various time steps and procedures of the analysis. The Modelbuilder constructs and stores the information for the structure being investigated. The element properties and constraints are defined in this module. The Analysis object is responsible for performing the analysis calculations, and is composed of multiple related sub-components for performing structural analysis operations. These sub-components are the Constraint Handler, DOF Numberer, AnalysisModel, SolutionAlgorithm, Integrator, SystemofEqn, and Solver. The recorder object monitors preset parameters in the model during the analysis and puts this data into a set of user-defined files. This module is important for documenting the behavior of the system as the analysis is performed.

A.3 Programming Strategy

OpenSees has been created using object-oriented programming strategy with the C++ programming language to form the objects necessary for the modules to operate. C++ is used as the programming language since it is the most widely used object-oriented language available, and thus has a significant support base. Generally, object-oriented programming involves the creation of objects composed of various modules that interact

within a set framework.

A framework can have various independent object classifications that have similar behavior and thus share various procedural methods, or functions. Implementing object-oriented techniques when constructing the OpenSees framework results in a final product that is capable of manageable source code changes.

The programming strategy used in the creation of OpenSees emphasizes the use of data abstractions within the programmed objects. Data abstractions represent a decomposition of data, algorithms, or calculations into the simplest of procedures to represent the essential behavior of the related data. Decomposition of the data and algorithmic processes required of analysis software is crucial in controlling complexity and ensuring a user-friendly environment for changing the software. Good abstraction in the base-levels of programming lead to modules that operate as independently as possible, and thus become easier to add to or modify within the framework of the program. This allows new code to be implemented into the code with as few changes to other objects or modules as possible.

A.4 Preprocessing

The majority of the preprocessing in OpenSees occurs in the form of text input files. These files interact with the OpenSees code through a scripting language and are executed by the OpenSees program interface. The OpenSees interface is an extension of the Tcl/Tk scripting language. The Tcl/Tk scripting language is free to download with the OpenSees source code. Information on the abilities of the scripting language and on downloading it may be found through the OpenSees website or online at www.scriptics.com.

The application of Tcl/Tk creates a versatile interface with the analysis package where the user can communicate with the program through scripting language routines. This becomes useful in the implementation of loops, calculations, and advanced analysis schemes. The OpenSees interpreter adds commands to the Tcl language for finite element analysis. Each of these commands is associated with a specific C++ procedure that is provided in the program code, and which may be altered according to a specific

development purpose (i.e. introducing new elements or materials). The associated C++ procedure is called upon by the interpreter to parse the Tcl commands from the text file and ultimately perform the entire analysis in sequence.

A.5 Modeling Capabilities and Overview

OpenSees is capable of creating complex structural models through the combination of simple input commands for a desired structural system. These commands are combined in the Modelbuilder object to create the physical system of interest. Typically, text input files can be created containing all of the commands necessary to create a multitude of structural systems. A wide variety of materials, elements, sections, and loads makes the program capable of evaluating diverse model types. Models may be created using 2, 3, or 6 degrees of freedom per node. Since every parameter is user-defined, there is no default set of units. The user must be certain to keep the units consistent while constructing the model and interpreting the results. Making use of the modeling capabilities available through the OpenSees interface makes modeling of structural systems possible.

Along with the OpenSees interface and commands built into the program, using the Tcl script commands affords the user a level of versatility beyond most program interfaces for model creation. Commands such as switches and loops can be formed to create various routines for parametric evaluation of a number of different model types under the same analytical parameters (analysis setup, loads, etc.). A level of programming experience and knowledge is required; however, the logic required to create relatively sophisticated systems is easily learned through examples. By referencing files together, file schemes may be created in order to keep each of the modeling sections separate and organized before the analysis is performed. Also, by defining variables for each of the modeling parameters, troubleshooting of the physical model is greatly simplified. Together, the program commands along with the Tcl interface provide a versatile and powerful analysis environment for model creation.

To create a structural model, a nodal array must be formed in the input file

describing each of the connecting points within the model. These are defined using the coordinates where the user wishes the nodes to exist. Nodes may be used to define the location of mass within a structure, and elements can be defined between two nodes. A variety of elements can be input as required by the model, such as trusses, beam-columns, and zero-length elements.

For the relationships to work between the created system of nodes, masses, and elements, constraints must be formed to govern the boundary conditions for the structure. A constraint is used to prescribe the value or relationship for a degree of freedom (DOF). Constraints can be formed for nodes to define the external support types (pinned, fixed, etc.) and to define multi-point constraints between internal objects for the model. If a zero-length item is created, for instance, a two-point constraint must be defined to govern the behavior of the two nodes between which this element exists. These commands are essential for establishing a stable and solvable system of equations for the model, and for enforcing simplifying behavioral assumptions.

OpenSees contains an extensive array of available material behaviors. Some materials are capable of simple linear or bilinear behavior, while others incorporate more complex behavior such as isotropic hardening, pinching, degrading stiffness, and prescribed hysteretic relations. Each material must be chosen based on the relevant parameters and how they relate to the desired material behavior. Many materials represent a variety of uniaxial stress-strain relationships. Also, materials are available to represent the stress-strain relationships at the integration points of a continuum for force-deformation elements. Materials such as these allow for very detailed model formulation and evaluation.

The creation of section elements may become useful for more detailed modeling of structural sections. The section command is used to create a section force-deformation object, which represents the force-deformation relationships at beam-column or plate integration points. The number and type of integration points is dependent on the section type chosen and the desired accuracy of the model. Elastic sections are defined by material and geometric constraints. A resultant section may be used to model a nonlinear force-deformation response. Fiber sections, for instance, may be used to discretize a cross-section into subregions called patches. Usually, fiber sections are

associated with reinforced concrete elements and the internal subregions create locations for the inclusion of reinforcement. The geometric parameters of a section must be defined with respect to a planar local coordinate system, and then transformed later into the global system.

Transformation of the local coordinate systems for specific elements and sections into the global system is achieved using the geometric transformation command. This object is used to transform the structural parameters for stiffness and resisting force into a single global system. The available transformation commands may be used to create rigid offsets applicable to panel zone models. P-Delta transformations are used to incorporate the second-order load effects into the model, and the Corotational transformation is useful for transformation within large displacement systems. Elements require the assignment of a transformation type in the input command.

The final piece for creating a model of a structural system in OpenSees is the application of loads and load patterns to the structure. Loads are created through a combination of pattern type, time series type, and actual load application. This allows for any combination of load type and pattern to be applied within a model. The pattern type must be defined first for either plain loads or excitation patterns (earthquake loading). Then, the time series over which the loads will act is defined, followed by a listing of nodes/elements and the applied loads. Typically, more advanced loadings such as earthquake ground accelerations are defined first as a path time series variable that references the ground motion file as the path. This variable can then be read into the argument for a uniform excitation pattern and applied in any direction to the model. Routines are available for the formatting of typical ground motion files into the format and file type useable in an OpenSees analysis.

A.6 Damping Types

Dampers may be added in OpenSees through the creation of an element with damping properties. There are two options for assigning properties to an element to represent a damper. The first option is to create a parallel material relationship with both viscous and stiffness properties of a damper and assigning it to an element. The damping

coefficient can be calculated based on the desired percent of critical damping. The second option is to assign Rayleigh damping parameters to structural elements on a region basis. A region can be defined as one or more elements in a structure, and the Rayleigh damping parameters for stiffness and/or mass proportional damping can be calculated based on the desired percentage of critical damping. Both methods have been proven to give consistent and reliable results in the verification models.

A.7 Output Capabilities and Postprocessing

Output may be chosen on a single response basis through the Recorder object in OpenSees. Each response can be recorded and sent to a specific text output file individually along with time step information (if desired). Each output creates a new output file containing the individual response results for an analysis, and an analysis can create any number of desired output files. There are multiple recorder commands available to record structural responses, and they include node response, envelope node response, maximum node displacement, drift between nodes, element response, and envelope element response. Recorders are typically created to monitor a specific response from either an element or node. Envelope recorders are available for recording only the maximum and minimum values of a specified response measure. This scheme of data recording is beneficial in the sense that only the specified results will be obtained without the inclusion of any extraneous data.

Limited postprocessing is available in OpenSees for dynamic analysis beyond the creation of data files for single response behavior (i.e. displacement, velocity, and acceleration). Some graphical windows and TclVideo objects may be created through the Recorder object, but these are only for viewing the data obtained by the recorder commands. For dynamic structural analysis, a large amount of Postprocessing is required for interpretation of the relevant data, such as damage measure calculation or graphical processing of a system in motion. Due to the large amount of data created by OpenSees in dynamic applications, iterative schemes can be created using the scripting language for keeping the data organized based on the user-defined parameters. However, a large amount of data may be compiled, and can later be used in conjunction with various

programs for the desired processing.

Currently, some graphical interfaces are being developed for the postprocessing of OpenSees data. These interfaces are being developed on a project by project basis, and thus they are difficult to apply to all-encompassing applications. One of the most promising ones is OSP, a Graphical User Interface created by Charles Chadwell (2001) for viewing input and result files. It contains an editor where linked input files can be viewed and edited, then seen graphically. Other programs such as Open Studio by Jaesung Park (2003) are under development as all-encompassing graphical interfaces and will facilitate the use of OpenSees for future users and applications.

The use of macros in Microsoft Excel is also an option for the sorting and analysis of large quantities of data. Macros may be created to read the data from the raw text files, plot the data, and perform any post-processing calculations required. Maximum values can be filtered and saved for the purposes of plotting IDA curves. These macros are helpful for viewing system data and response histories; however, the macros may become complex to create. The Excel spreadsheets can only handle as many as 65,536 data points in sequence, thus limiting the number of data points that can be processed without using multiple columns or files. Another macro can be created to trim a full acceleration response record down to a lesser time step so that the data is easier to manage in Excel.

A.8 External Features

In order to keep all of the program options and commands straight, some reference materials are available through the OpenSees website[°]. Examples are available for input file creation, as well as documentation for various program topics. Perhaps the most important document is the OpenSees user's manual which provides a basic reference for the program commands. Because the program is constantly under development, the manual is constantly being developed as well. However, there is an online version that can be downloaded periodically. The manual gives descriptions of the command lines used to perform the functions of the program, and is useful in getting to

[°] OpenSees website is located at http://opensees.berkeley.edu/

know how to communicate with OpenSees.

Also available is a Tcl editor for use in creating text files and which provides error checking, a command index, and direct access to the program interface to analyze a model. This is useful in keeping multiple files organized through one interface and to facilitate model creation and execution. Along with this editor, the example model constitutes a comprehensive primer for getting started with OpenSees.

A.9 Modifying the Source Code

Perhaps the most important feature of OpenSees is the open source nature of the system code. Anyone who wishes may download the latest version of the program code and re-build it with any changes which may be desired. Any part of the code may be changed depending on purpose, from adding new materials and elements to troubleshooting the existing code. The current program version may be compiled on a wide range of platforms (UNIX, Windows, Mac) using the C++ programming language.

Currently, the available code from the OpenSees website is compatible with the latest version of C++ found in Visual Studio.Net. After downloading the source code from the website, this version of C++ must be present on the computer along with the accompanying version of the Tcl scripting language program applicable to the release of the code being built. Further details are available on the website regarding the presence and location of system files required for accurate compilation of the code. Simply copying the correct file folders to the correct directories on a computer takes care of this requirement.

All previous versions of the source code are available for download along with the “developer’s release”, which is the latest source code release of the program. This is different from the OpenSees program executable, which is downloaded and used to run OpenSees for analysis. Both the source code and program executable are re-released periodically as new additions are made to the code. However, the source code release may not be inclusive of all options available in the latest OpenSees program executable.

Users may submit the latest code additions to the source code though the use of Concurrent Versions System (CVS). After enough new development has occurred, new

releases of the source code and program executable are made available approximately once or twice yearly. Updates to the source code and the executable do not always occur simultaneously, resulting in discrepancies between the available features contained between the two.

Once the code has been modified according to a user's purposes and successfully compiled, these changes may be submitted, or checked-in, for inclusion in future releases of OpenSees. Submission of code additions occurs through CVS as well, creating a constant dialog between the program users and program creators. For security purposes, submitting new code is not as easy as downloading the latest additions. Check-in procedures are monitored by the program administrators, whereas the check-out procedure requires only a username and password. This process results in a constant dialogue between program users and developers that modifies the code into an increasingly powerful analytical tool. This unique code strategy increases the applicability of the OpenSees program and encourages additions to the code.

The OpenSees source code has been compiled with the addition of new elements and materials for research at Virginia Tech. Elements that consider shear deformation are useful in multiple areas of analysis, as well as materials that behave with prescribed properties that are applicable to structural devices. The new material addition behaves elastically along a nonlinear stress-strain relationship, or hyperelastically. These additions to the code have been developed in the C++ language as an elastic material where the stress-strain behavior can be defined using a cubic polynomial expression. The input required of the user is the polynomial relationship input in the command line for this material as a sequence of coefficients and exponents. These new materials and elements are expected to be useful in nonlinear dynamic structural analysis using OpenSees. More details on the programming of the new material behavior and the verification of its behavior may be found in section B.3.1 of Appendix B. The C++ Source code files for the new additions in OpenSees may be found in Appendix C.

A.10 OpenSees Limitations

Due to inexperience with, and the newness of, OpenSees, there are still plenty of

limitations regarding the use of this program. While documentation is available, the lack of an up-to-date and broad reference document makes learning the advanced features difficult. The user's manual has been compiled more as a quick reference than an in-depth guide to the modeling and analytical capabilities of the program. This requires the user to use trial and error methods for learning and troubleshooting rather than productively learning from a single comprehensive source. Also, the constant development of new program capabilities requires a constant learning process for the code that the current documentation is not capable of maintaining.

Another limitation of OpenSees that makes troubleshooting and result interpretation difficult is the lack of a graphical user interface (GUI). Most structural systems are best described in graphical format, making text files difficult to interact with on a user level even through a useful scripting language. There are some GUIs being developed for use with OpenSees, however they are being developed as tools for specific projects and therefore lack the scope for broad applications.

Also, due to the inexperience of OpenSees in multiple applications, a library of reference material has yet to be made available for model verification. Traditionally, as programs are developed, typical model types are built and verified versus established results from other programs and analyses. As OpenSees becomes more widely used and applied, this problem should be solved with the correct networking of files. Currently, however, this necessitates an extensive evaluation of the program to verify accuracy and to validate modeling techniques.

A.11 Conclusions

OpenSees is a fully capable and powerful analysis tool that requires more developmental attention before it can reach its full potential. The ever-evolving nature of OpenSees has created a new paradigm for structural analysis where the capabilities of the software are essentially unlimited. Future applications for the program include use in pseudodynamic testing, nonlinear dynamic analysis, soil-structure interaction, and parametric evaluation of complex systems. The current capabilities of OpenSees will prove to be both adequate and capable for use in the proposed nonlinear dynamic analysis

of structures with hyperelastic devices. As more capable user interfaces are created and the learning process for the software is refined, more users will be introduced to the analytical potential of OpenSees. The learning curve may be steep for current users, but the capabilities of Opensees will be greatly enhanced as user-friendly interfaces are added to the program features.