Hindawi Publishing Corporation
Modelling and Simulation in Engineering
Volume 2008, Article ID 937542, 10 pages
doi:10.1155/2008/937542

Research Article

Open-Source Software in Computational Research: A Case Study

Madhava Syamlal,[1] Thomas J. O'Brien,[1] Sofiane Benyahia,[1] Aytekin Gel,[2] and Sreekanth Pannala[3]

[1] *National Energy Technology Laboratory, P.O. Box 880, Morgantown, WV 26507, USA*
[2] *Aeolus Research Inc., 18 Cecil Drive, Dunbar, PA 15431, USA*
[3] *Oak Ridge National Laboratory, Building 6012, MS-6367, RM-101, Oak Ridge, TN 37831, USA*

Correspondence should be addressed to Madhava Syamlal, madhava.syamlal@netl.doe.gov

Received 11 February 2008; Accepted 11 March 2008

Recommended by Andreas Tolk

A case study of open-source (OS) development of the computational research software MFIX, used for multiphase computational fluid dynamics simulations, is presented here. The verification and validation steps required for constructing modern computational software and the advantages of OS development in those steps are discussed. The infrastructure used for enabling the OS development of MFIX is described. The impact of OS development on computational research and education in gas-solids flow, as well as the dissemination of information to other areas such as geophysical and volcanology research, is demonstrated. This study shows that the advantages of OS development were realized in the case of MFIX: verification by many users, which enhances software quality; the use of software as a means for accumulating and exchanging information; the facilitation of peer review of the results of computational research.

1. Introduction

Open-source (OS) software is ubiquitous; knowingly or unknowingly, the present reader is likely to be making use of OS software embedded in a gadget or device while reading this paper. Just as in software applications, OS software is making an impact on computational research as well. In this paper, we examine the role of OS software development in computational research in engineering. It is not certain that the advantages of OS development demonstrated in application software such as Linux and Apache can be realized in the case of computational research software. To address that question, we present a case study of the computational research software Multiphase Flow with Interphase eXchanges (MFIX). Since 2001, the source code of MFIX is being distributed to users that register at the MFIX website. MFIX license, however, does not strictly conform to currently accepted standards for OS licenses. Nevertheless, the license has clearly allowed the advantages of OS development to be studied (see the discussion in Section 3.1). (MFIX license is expected to be changed into a form similar to MIT open-source software license.) We discuss how OS development facilitates software verification, model validation, peer review of computational results, and the accumulation and exchange of information in computational research.

The term open-source software came into existence in 1998 although such software has existed from the early 1990s [1, 2]. Its main feature is that users have access to the source code, and they may study or augment the code to change the software's functionality. One year before this term was coined in 1997, a lesser-known term, *Bazaar* development process, was proposed to label the community style software development facilitated by the internet. Users can have access to the source code and also watch and contribute to the development of the software. This development process has revolutionized software development, as exemplified by highly successful software products such as the operating system Linux and the web-server Apache.

Research communities, for example, in radiology [3, 4], as well as drug discovery and bioinformatics [5, 6] have been early adopters of this development process. The economic and motivational questions, promises, and pitfalls surrounding OS development are being debated in archival journals, for example, [3–11]. Surprisingly, to the best of our knowledge, nothing has been reported in engineering journals about the role of OS development in computational research, which is the topic of this paper. Because of the smaller number of researchers involved in computational science and engineering, not only the effectiveness but also the sustainability of the OS development process is in question.

The discussion in this case study pertains only to software used for computational research; that is, software used for the development of mathematical models and numerical techniques. Our discussion does not pertain to the end-use of engineering software for the analysis and troubleshooting of devices used in industry or in a laboratory. In those cases, the simulation results are used for designing or troubleshooting a device and neither the mathematical models nor the numerical techniques underlying the software per se are the object of the investigation.

For the case study, we use the gas-solids flow software MFIX (http://www.mfix.org/) developed at the National Energy Technology Laboratory (NETL, WVa, USA). The information on user experience was collected from two user surveys, which enabled us to determine who used the software, how it was used, how successful the users were, and what were the significant outcomes. The information from those surveys and the MFIX development team is reported here to illustrate how well the OS development process worked in the case of MFIX.

We start with a brief overview of the OS development process in Section 2 and discuss the advantages and disadvantages of the OS development process in the context of verification and validation. The background and OS features of MFIX are described in Section 3. Section 4 describes the OS experience with MFIX: the software's impact on research and education in computational gas-solids flow and the contributions to MFIX from external users. The conclusions are given in Section 5.

2. Open-Source Development

A number of computational research codes have been always freely available in a source code form (e.g., NETLIB repository, TEACH code from Imperial College, KFIX, and KIVA from Los Alamos National Laboratory). The distinguishing feature of the OS label is that the software is provided with a license [7, 10] that requires users to follow some simple rules such as the inclusion of a license header in each source file. This type of licensing played a crucial role in making the OS development process successful.

This new way of developing and maintaining software with the involvement of users has made a big impact in the world of business software. Raymond [12] calls the new process the *Bazaar* model and contrasts it with the traditional *Cathedral* model used by commercial software developers. In the *Cathedral* model, a few experts develop the software and make every effort to release it only after fixing all the bugs, but users are not involved in the development process. In the *Bazaar* model, users may watch and participate in the code development process. Raymond [12] credits Linus Torvalds for the invention of this model and claims that bugs will be discovered at a rapid rate in this model because the source code is available for public testing, scrutiny, and experimentation. In contrast, an inordinate amount of time is spent by a few developers hunting down bugs in the code being developed under the *Cathedral* model. Raymond predicts that although individual vision and brilliance will matter during the initiation of software projects, software will increasingly be developed using the open-source development process. In this paper, we will use the terminology OS development, which is synonymous to the *Bazaar* model, to mean the availability of the software in source code form as well as its development in full public view, promoting user participation.

All the evidence that supports the OS development process, however, comes from business software products, which are comparatively large (millions of lines of code) and have a large number of users (thousands). Even in that case, the claim that OS development is "thebest" approach for software development has been disputed [8]. The advantages of OS development of the relatively smaller (thousands of lines of code) computational research software with a smaller user base (tens or hundreds) are uncertain. This study shows that the advantages of OS development have been realized in the case of MFIX, and shows the additional benefit of facilitating the peer review of computer code used for generating results that are published in journal articles.

2.1. Verification and Validation

The accuracy of computational software is ensured through a verification and validation process. As defined by the American Institute of Aeronautics and Astronautics (AIAA), verification is "the process of determining that a model implementation accurately represents the developer's conceptual description of the model and the solution to the model" and validation is "the process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model" [13].

The verification of complex software is a crucial and laborious endeavor. It is also an ongoing process when the mathematical models and numerical techniques are continually being improved, as in the case of the gas-solids flow models in MFIX. Every time the mathematical model or the numerical technique is changed, bugs may be introduced. The advantage of a large community of users is that bugs may get detected quickly. This is an advantage readily enjoyed by commercial software. The user base of OS software must reach a critical mass before the OS software can realize the advantage. Once a critical mass of users is reached, then OS software offers the additional advantage that the user has the ability to identify and fix bugs in the source code in addition to merely reporting them as in the case

of closed-source software, thereby reducing the burden on the developers. Furthermore, OS software allows the users to conduct code review even without being prompted by a bug. This happens when a user reads the code to learn the implementation details of a particular equation. This advantage of OS development cited by Raymond [12] cannot be realized in closed-source software development because only a few developers are able to review the code.

2.2. Peer Review of Computational Results

Unlike verification studies, validation studies of models are of much scientific interest and get reported in journal articles. OS development offers two advantages for the peer reviewing of published computational results. First, because OS software is freely available, a greater number of researchers have the opportunity to reproduce published results, which allows researchers to better understand and contribute to the shared information. Researchers routinely check mathematical derivations presented in research papers, and experimental findings are usually checked for reproducibility. Similarly computational scientists frequently try to reproduce published results, and OS development facilitates that by making the research software universally available.

Second, the more important advantage of OS software is that the peer reviewer has access to all the implementation details of the software. While comparing published computational results, questions may come up regarding the mathematical model (e.g., the equations, boundary conditions, constants) and computational algorithm (e.g., how certain limits are evaluated). Even though the mathematical model is usually fully stated in the paper, being able to "read" the model equations from the code is an advantage. For example, typographical errors in two equations in [14] could be detected by reviewing the corresponding MFIX code [15]. The advantage is even greater in the review of the numerical algorithm because its complete description is often not reported in papers. Although papers describe the main algorithm in detail, certain seemingly secondary details may not be reported. In fact, certain algorithms in commercial software may even be guarded as proprietary information. In the absence of such details sometimes it is difficult to establish whether a difference in the solutions given by two codes results from the differences in the underlying mathematical models, the numerical algorithm, or the code implementation of the numerical algorithm. The availability of the source code allows readers and reviewers to unequivocally settle such questions about the mathematical model, the numerical algorithm, and the code implementation.

The advantage discussed in this section is unique to software used for computational research. When the results of such studies are reported in journal articles, the readers are interested in the details of the mathematical model and numerical algorithm used in the software. This is not the case with typical OS software, and, therefore, this advantage is not discussed in the literature on OS software (e.g., [12]).

2.3. Barriers to OS Development

Although OS development offers advantages for software verification and validation and peer review of published results, it may have several disadvantages when used for computational research. One, for its existence, OS software needs a core group of researchers supported over a sufficiently long period of time, which is not always feasible with OS software as the main objective. Two, for realizing the advantage of community verification and validation, the user community must grow to a critical size. This is hard to achieve because research software often has only a limited number of features, which may interest only a small number of users. Also, attracting developers is difficult because of the unique expertise required and intellectual property issues involved [5]. Three, to develop software of sufficiently high quality attractive to external users, the OS team needs to follow good software development processes. For example, Gambardella and Hall [9] show that it is important that "lead" researchers establish a norm of contribution. A poorly written code does not attract users even when the source code is freely available. Writing readable and maintainable code, however, is not the highest priority of researchers who are usually trying to solve a problem within tight time constraints. Four, the software and its usage needs to be documented, which is often not done because scientists do not get credit for published OS codes [16].

If researchers realize the advantages of OS software and institutions begin to value OS software contributions, then the above difficulties could be overcome. In fact, that is the recent trend in government-funded computational research. In 2000, the Presidential Information Technology Advisory Committee recommended that "The Federal government should encourage the development of open source software as an alternate path for software development for high-end computing." Also, government agencies such as the US Department of Energy and National Institutes of Health are now encouraging the OS approach in several projects funded by them [3].

3. The Open-Source Code MFIX

We use our experiences with MFIX to examine whether the advantages of OS development are borne out by experience in the case of typical computational research software. Before presenting the case study information, we will discuss the background of MFIX and its OS features.

3.1. Background Information

MFIX development started at NETL in 1991. The main goal was to develop a tool for modeling fluidized-bed reactors such as coal gasifiers, commonly used in fossil fuel plants. The first version was completed by January 1993. All the variables used in the code were described in comment statements; the equations documented in a theory manual [17]; the numerical technique, code architecture, and user instructions documented in a user's

manual [18]. The first set of gasifier simulations were conducted in 1995 [19]. The public distribution of the code through the Energy Science and Technology Software Center (http://www.osti.gov/estsc/) started in 1995.

MFIX is a general-purpose computer code for describing the hydrodynamics, heat transfer, and chemical reactions in fluid-solids systems. The code is now routinely used at NETL for gasifier modeling, for example, [20]. It solves a generally accepted set of partial differential equations for the conservation of mass, momentum, species, and energy for multiple phases, for example, [21]. A summary of the balance equations is given in [22]. MFIX has been used for modeling bubbling, circulating fluidized beds, and spouted beds. The calculations give transient data on the three-dimensional distribution of volume fractions, pressure, velocity, temperature, and species mass fractions. Simulations are set up using an input data file, and in some cases, user-defined subroutines. Two post-processing codes are used for visualizing the results and extracting data in the form of tables from the output files.

The code underwent several revisions since 1995. The numerical technique used in the code was changed and several high-resolution discretization methods were added in 1996 [23]. In 1998, the code was translated from Fortran 77 to Fortran 90 using the translation software VAST/77to90 developed by Pacific-Sierra Research Corporation. Subsequently, the code was parallelized to run on shared-memory processors (SMP), distributed-memory processors (DMP) [24], or hybrid SMP/DMP machines [25]. In 1999, version tracking of MFIX was started by using the concurrent versioning system (CVS) software. Today, MFIX consists of 118 000 lines of Fortran 90 code, organized into 508 files and 969 subprograms.

A website, http://www.mfix.org/, was launched in 2001 for distributing the source code and disseminating information related to computational gas-solids flow. By August 2006, over 1 000 researchers from 250 research institutions worldwide had registered at the website. The website typically receives between 5 000 and 10 000 hits every month.

The MFIX license at the time of collecting data for this paper allowed registered users to download and modify the source code; however, no redistribution of the code was permitted other than through the MFIX code repository. Because of the registration requirement and the restriction on redistribution, MFIX license did not satisfy the OS definition specified by Open Source Initiative (http://www.opensource.org/docs/osd/). (It is expected that MFIX license will soon change to a license similar to MIT Open Source license.) Arguably, the lack of ability to freely redistribute the code could have diminished the users's enthusiasm to use and contribute to MFIX. Nevertheless many users signed up to use, review, and improve the code. The data presented here shows that the advantages of OS distribution were realized, which is perhaps a conservative assessment of OS distribution because a standard OS license could have increased the user response.

3.2. Infrastructure for Open-Source Development

Today, typical OS projects are supported by popular sites such as (http://sourceforge.net/index.php). SourceForge was a little over a year old when MFIX became OS in 2001, and the MFIX team decided to create its own OS development infrastructure. Not using a well-known OS website reduces the visibility of the project among OS developers. However, since MFIX is a computational research code, its potential users and developers are effectively reached through technical conferences and research papers.

Another feature of typical OS development environments is a bug tracking system, which has a user-interface for reporting bugs and a database for storing information on the bug. This feature is missing from the MFIX infrastructure, and bug reporting and resolution are handled with the mfix-help mailing list.

The MFIX infrastructure consists of seven components, which are described below in the order of their importance.

(1) *Downloadable code.* Three versions of the code can be downloaded from the website.

(i) CVS version. The users can view the latest changes in the code and download the desired files. This ensures that the development is done under full public view and all the modifications are immediately available to users.

(ii) Development version. Every day the previous day's version of the entire code is gathered and placed on the website, making it convenient for the users to download the latest version of the code.

(iii) Stable version. This version, released once or twice a year after completing a major revision or several minor revisions and bug fixes, is tested and guaranteed to work with all the test and tutorial cases.

(2) *Source code revision control.* CVS (http://ximbiot.com/cvs/) is used for source code revision control. CVS allows several developers to work on the same file at the same time and merge their revisions of the code. CVS records all the changes in the code; all previous versions of the code are archived in the database. Therefore, developers can easily retrieve any version of the code by specifying the desired date and time. A version referred to in a publication is forever available for public scrutiny (provided the authors preserve user-defined routines, if any). With the CVS web interface (http://www.freebsd.org/projects/cvsweb.html), users have immediate access to the changes in the source code. The web interface gives the file name, version number of the file, age of the file, name of the developer that checked in the latest version, and comments about the changes made. The interface allows users to graphically compare two versions of the files and pinpoint the differences. All users can check out code from CVS, but the main developers act as gatekeepers for checking code into CVS.

(3) *Tests and tutorials.* The MFIX directory contains over 40 test cases and 13 tutorial cases. The test cases verify various features of the code in isolation (e.g., fluid flow without solids), and typically run in a short period of time. When a new feature is added to the code, a new case to test that feature is also added to the test-cases directory. The

tutorial cases test a combination of features of the code (e.g., combined fluid-solids flow) and can be used to learn about setting up simulations.

(4) *Documentation*. The documentation section provides detailed manuals [17, 18, 23] and developer notes, written for each new feature implemented. It is impractical to synchronize the detailed manuals with a code that is constantly evolving. So, the MFIX team maintains two minimalist manuals that are continually synchronized with the code: a *readme* file that lists all the user inputs and an *MFIXequations* file that lists the current set of equations [22]. The equations file is a version-controlled, citable web document. Also, the code is internally documented with comment lines, which constitute about 62% of the code.

(5) *Communication*. The communication between users is facilitated with several mailing lists based on the OS software SYMPA (http://www.sympa.org/). The main mailing list mfix-help@lists.mfix.org is where users post help requests. All the discussions are archived, which the users can search to find past discussions about a certain topic. In addition to the mfix-help mailing list, over 20 different mailing lists promote discussions on specific aspects of multiphase flow: numerics, granular physics, discrete element methods, chemical reactions, visualization, validation, applications such as gasification, and so forth.

(6) *Test harness*. A test harness, powered by the OS testing environment QMTest (http://www.codesourcery.com/qmtest), is used to conduct software regression tests. Such testing seeks to uncover regression bugs or broken software functionality that was previously working. Typically, regression bugs occur as an unintended consequence of code changes. The test harness checks out the current version of MFIX every night, builds executables, and runs the test cases identified by the developers. The results of the tests are posted on the MFIX website. If any of the tests fail, then the development team is alerted with an e-mail. Such early detection of bugs reduces the debugging effort because the bugs can be readily correlated with the previous day's changes. This ensures the quality of the evolving software on a daily basis.

(7) *Open citations* (http://www.mfix.org/opencitations/index.php). This section of the website lists papers relevant to computational gas-solids flow. The discussion board allows researchers to submit information and comments on papers in computational gas-solids flow. The database is searchable by categories such as author name and subject. This part of the website has not developed very well perhaps because it has not been adequately publicized.

4. A Case Study on the Effectiveness of the Open-Source Approach

MFIX was made open source primarily to make it available to a large number of researchers and, thereby, to supplement research articles based on MFIX by exposing all the implementation details. MFIX was already being used by a few external users who got access to the code by contacting NETL. Also, MFIX team was planning to set up a webpage for internal collaboration, and making the webpage and the code available to external users was an easy next step. A secondary reason was that the team expected to derive the benefits of OS development in verification, validation, and code enhancements. In this section, we will examine how well these expectations were fulfilled. First, we will describe how MFIX has contributed to research and education in computational gas-solids flow. Then, we will examine how MFIX itself was enriched because of the OS development process.

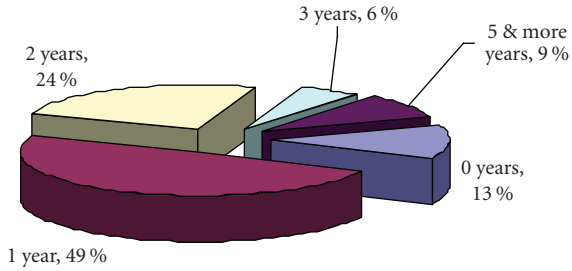

Figure 1: Distribution of survey respondents by years of experience.

To collect information on MFIX usage, two surveys were conducted. The first survey was conducted in 2005, and input was solicited from all MFIX users by sending an email message to mfix news list [26]. The survey was conducted to develop guidance for the future development of MFIX. A second survey of a selected small group of researchers was conducted later to collect information on significant outcomes from the use of the software.

4.1. Impact on Research and Education

The first survey resulted in responses from 70 users, about 10% of the registered users, with wide-ranging backgrounds and application needs. Figure 1 shows the distribution of MFIX users based on the number of years they have been using MFIX. The majority of the users have around one-year experience, whereas 30% have two to three years of experience; no one had four years of experience. The most experienced users (five or more years of experience) constituted about 9% of the user community, which is reasonable because at the time of the survey, MFIX had been an open-source code for only four years. Even in the future, we expect that many users will remain in the category 1–3 years of experience because many of them would be graduate students.

Figure 2 shows the affiliations of MFIX users: 41% are graduate students, 21% are faculty members, and 16% are postdoctoral associates. Therefore, almost 80% of the users are from universities. Only four respondents identified themselves as industrial researchers (excluding MFIX team members) and another as an attorney. We believe that this is a representative sample based on our experience with the requests for help although the survey results could be biased because all industrial users may not have responded.

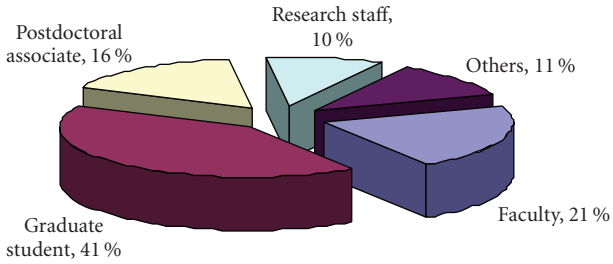

Figure 2: MFIX user categories.

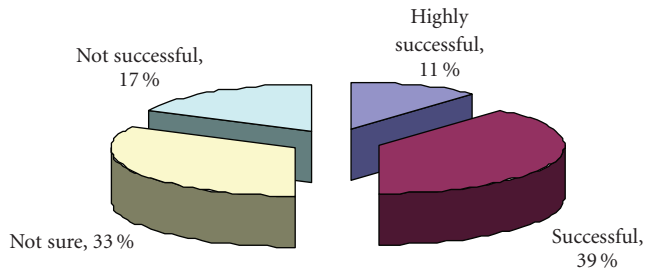

Figure 3: Success rating in using MFIX.

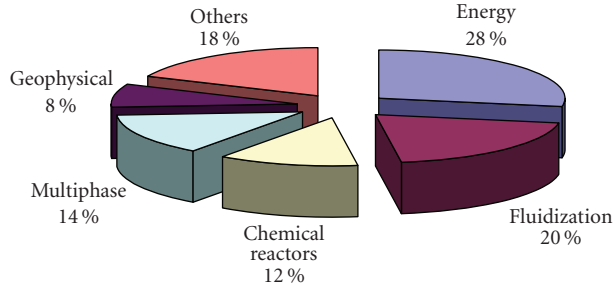

Figure 4: Different categories of MFIX usage.

Figure 3 shows the percentage of MFIX users that consider the use of MFIX in their projects is being successful. Nearly 50% said that they had success with their MFIX projects, among which 11% said that they were highly successful. The other half was split between being not sure (33%) or not successful (17%). The percentage of successful users is encouraging because MFIX is an expert user's software with minimal user support and no training provided. About 33% of the users reported "not sure" presumably because they had recently started using MFIX. The response "no success" in some cases may mean that the model results did not agree with experimental data because of a deficiency in the mathematical model, which in the context of the present discussion would be a successful application of the software. Most of the users who indicated no success had used MFIX for less than a year and described their expertise level as "Basic." On the other end of the spectrum, users who indicated high success had been using MFIX for 3-plus years and described themselves as advanced users who were comfortable making changes in the source code.

Figure 4 shows how the respondents used MFIX in their research. The responses that were blank or not specific (e.g., "general") were discarded; the responses that could be counted in multiple categories were counted in all the applicable categories. All told there were 65 responses. Nearly half of the applications are similar to the application for which MFIX was originally developed: 28% in the category Energy (coal gasification and combustion, biomass combustion) and 20% in the category Fluidization (bubbling fluidized beds, risers, particle flow, gas-solids flow). About 12% of the applications are in the related category of Chemical Reactors (fluid catalytic cracking, fluid bed reactors, and polymerization), and 14% in multiphase flows (multiphase microfluidics, slurry flow, gas-liquid). Interestingly, the use of MFIX has migrated to other application areas; Geophysical (volcanic granular flows) has become a distinct category with 8% of the responses and other applications (microchannel heat exchanger, powder flow) accounted for 18% of the responses. We could not assess certain other uses of code stated in the registration form (e.g., to check how a certain algorithm is implemented) because such incidental users did not respond to the survey.

A second survey was conducted to collect detailed information from 11 principal investigators who had been known to be significant users of MFIX at that time. This sample was selected based on researchers that had frequently contacted the MFIX team for information. They were asked to provide a brief description of the problem being solved, significant outcome (e.g., graduate thesis, papers, improved algorithms or theory, migration of code/algorithm into other software, applications, improvements to design), and to comment on their experiences with the OS development process.

The responses regarding the research topics and significant outcome are summarized in Table 1. Also, several groups not included in the survey have published research results using MFIX, for example, [27–30]. In the last five years, there have been over 50 publications and presentations and 15 graduate theses that are based on MFIX. These demonstrate that the OS availability of MFIX has significantly contributed to research and education in gas-solids flow. The advantage of model validation by a community of users, in addition to the MFIX team, is being realized.

4.2. User Feedback

The responses to the question about what worked well for the user and what needs improvement are given in Table 2.

A user stated that "the open-source model has been very successful because the code is very well documented so that users can see where various tasks are assigned and how they interrelate, the MFIX team has managed to keep the process of code updates robust and well maintained, and the community of users and contributors are mostly already well trained in CFD and theories of multiphase flow" [50]. He emphasized that to be successful, "open-source approach needs some of the "care-taking" and oversight activities of the MFIX team, to manage where code improvements

Table 1: Research topics and significant outcome.

Research topic	Significant outcome
Theory development	(1) Modeled segregation in gas-solids fluidized beds. Introduced the effect of particle rotation using an effective coefficient of restitution. Compared with experimental data to demonstrate the effect of particle rotation on bubble dynamics [31, 32].
	(2) Studied the statistical properties associated with time-dependent, spatial inhomogeneities that occur in fluidized suspensions. Gathered fluctuation statistics and used this information to construct closure relations for filtered two-fluid models [33–36].
	(3) Simulated flows at high particle volume fractions, where frictional stresses domina [14].
	(4) Implemented cohesive forces into the discrete-particle framework using a square-well potential [37].
	(5) Studied segregation/mixing of dense binary mixtures in fluidized beds. Investigated the various driving forces for segregation, especially driving forces that arise from a nonequipartition of granular energy [38].
Numerical techniques development	(1) Developed direct quadrature method of moments (DQMOM) to simulate particle aggregation and breakage in a fluidized-bed [39, 40].
	(2) Implemented the algorithm *in situ adaptive tabulation* (ISAT) to solve complex chemistry calculations in a fast and efficient manner. ISAT technique speeded up a silane pyrolysis reactor simulation by a factor of 48 [41, 42].
Model validation	(1) Modeled elutriation of char from a bubbling fluidized bed; simulated the simultaneous elutriation and gas-solids reactions of char particles in gasifiers. Compared elutriation data with simulation results [43].
	(2) Simulated bubbling fluidized beds (Geldart Group A/B and B particles) and compared predictions with electrical capacitance tomography data. Identified limitations in the model predictions and determined their cause (no cohesive force; no frictional stress above a void fraction of 0.5). Proposed and tested modifications for solving the model limitations identified [44, 45].
	(3) Modeled dense-phase fluidized beds containing fine catalyst powder (e.g., FCC stripper) [46–48].
Model application	(1) Developed and validated a model of a polyethylene pilot-scale fluidized bed at Univation. The validated model was used to locate hot spots in a reactor [49].
	(2) Simulated (a) high-Reynolds number volcanic eruptions and associated multiphase gravity currents, and (b) low-Reynolds number chaotic convection in magma chambers [50].
	(3) Modeled air-gravity conveyors (airslides) in which the flow of the granular material is enhanced by the air that is forced through the bottom of the conveying trough [51].
	(4) MFIX-family codes used as quality-assured numerical tools to explore multiphase dynamics (e.g., dust explosions) in the Yucca Mountain Project drift/repository, Nevada, the proposed site for the first permanent geologic repository for high-level radioactive waste in US [52–55].
	(5) Developed the model of a solar receiver. Characterized the flow dynamics of a curtain of free-falling ceramic particles, heated by concentrated solar energy within an open cavity solar receiver to temperatures in excess of 900°C [56].
	(6) Simulated heterogeneous catalyses in microchannel heat exchangers using a porous body approach [57].
Train graduate students	Several graduate theses, for example, [42, 58–61] .

Table 2: User feedback.

Worked well	*Needs improvement*
(i) Open source access	(i) Ability to represent complex geometry
(ii) Well-written code	(ii) Continual changes in the code make it difficult to repeat/reproduce earlier work with the revised code
(iii) Well-organized website that is regularly updated	(iii) Documentation needs to be updated and made thorough
(iv) Discussion forum and archived messages	(iv) Undocumented capabilities that users become aware only after browsing through source code
(v) Questions are answered quickly, issues resolved quickly	(v) Tutorial that one can use to learn how the code is configured
(vi) Training of students at NETL in summer	

Table 3: Major capabilities added to MFIX by graduate students.

Capability	Contributor	University
Granular energy equation	K. Agrawal	Princeton U.
Frictional flow model	A. Srivastava	Princeton U.
Lees-Edwards boundary condition	P. Loezos	Princeton U.
DQMOM	R. Fan	Iowa State U.
ISAT	N. Xie	Iowa State U.
Discrete element model (DEM)	D. Boyalakuntla	Carnegie Mellon U.
Cohesion model in DEM	M. Weber	U. of Colorado.
SI units capability	S. Dartevelle	Michigan Technological U.
Koch and Hill drag correlation	C. Sutton	Lehigh U.

Table 4: MFIX bugs reported by users.

No.	Bug report	Verification	Contributors (affiliation)
(1)	Partial slip boundary condition	Code	K. Agrawal (Princeton U.)
(2)	Granular energy equation source terms	Code	S. Dartevelle (Los Alamos), J. Galvin (U. Colorado)
(3)	Cylindrical coordinate stress terms	Code	J. Galvin (U. Colorado)
(4)	Problem with Cp in the energy equation	Solver	U. Imke (Forschungszentrum Karlsruhe GmbH)
(5)	Inner radius in cylindrical coordinates	Solver	J. Pasini (Cornell U.)
(6)	Solid-body rotation problem	Solver	A. Srivastava (Princeton)
(7)	Species mass balance problem	Solver	T. McKeen (U. Saskatchewan)
(8)	Momentum deficit in cyclic simulations	Solver	A. Andrews (Princeton)
(9)	Compilation problems	Code	L. Oger (U. of Rennes)
(10)	Open-MP bugs	Code	S. Dartevelle (Los Alamos)
(11)	Uninitialized variables	Code	S. Dartevelle (Los Alamos)

are best implemented. Without this oversight capacity, the open-source model would falter and various versions of the code would emerge with questionable validation, etc." This comment resonates with the finding of Gambardella and Hall [9] that "without some kind of coordination, production of the public knowledge good (science or research software or database) is suboptimal" and that "if "lead" researchers are able to establish a norm of contribution to the public good, a better outcome can be achieved."

4.3. User Contributions

Having discussed the impact of MFIX on research and education in computational gas-solids flow, we will now show that MFIX itself was enriched because of the OS development process. Table 3 summarizes major contributions to MFIX code from external contributors who happened to be all graduate students at the time of making the contribution. In most cases, these developers sought the advice of the MFIX team, but the development itself was not done under CVS control. The code modifications were later merged with MFIX CVS by one of the MFIX team members.

During 2001–2006, mfix-help email list received 1575 messages of which around 20% reported bugs or bug fixes. Table 4 lists some of the significant bugs detected and reported by various external users. Some bugs (items 1, 2, 3) were caught through *code verification* (i.e., the users simply reading the source code.) So, the benefit of code verification through many eyes reading the source code [12] is being realized. Some bugs were identified during *solver verification* (i.e., comparing simulation results with known solutions), but the users then read the source code and reported the location of the bug (items 4, 5), again vindicating Raymond's claim [12]. In some cases, the problem was detected through solver verification (items 6, 7, 8), but the users could not identify the cause of the problem. The MFIX team later resolved these programming errors (item 6) or defects in the computational model (items 7, 8). Some bugs got exposed when the user employed a compiler different from that used by the MFIX-team (which can be considered to be another form of code verification). Some bugs were merely compiler quirks (item 9), some were applicable only to SMP mode of execution not often used by MFIX team (item 10), but some others were potentially of wide-spread impact (item 11).

5. Conclusions

A case study on the open-source development of a computational research code, MFIX, is presented here. Five years after the open-source development of MFIX was started

nearly 80% of the MFIX users are from universities and half of the users reported success in using the software. Half of the applications are similar to the applications for which MFIX was developed (coal gasifiers and fluidized beds), and, interestingly, the other half are in new areas such as geophysical flows. During the five years, the use of the software resulted in over 50 publications and presentations and 15 graduate theses. MFIX itself was enriched because of the OS development. Several major capabilities in MFIX were developed by external users, and the advantage of many eyes verifying the code was realized with several bugs being reported or fixed by external users. In one instance, access to the source code helped with the detection of typographical errors in the equations given in a journal article. From these experiences, it appears that the advantages of OS development were realized in the case of MFIX.

Acknowledgments

The submitted manuscript has been coauthored by a contractor of the US Government under Contract no. DE-AC05-00OR22725. Accordingly, the US Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so for US Government purposes.

References

[1] "Open-source software," August 2006, http://en.wikipedia.org/wiki/Open-source_software.

[2] "History of the Open Source Initiative," August 2006, http://www.opensource.org/docs/history.php.

[3] B. J. Erickson, S. Langer, and P. Nagy, "The role of open-source software in innovation and standardization in radiology," *Journal of the American College of Radiology*, vol. 2, no. 11, pp. 927–931, 2005.

[4] D. P. Harrington, "Open-source software opportunities and challenges," *Journal of American College of Radiology*, vol. 3, no. 1, pp. 14–15, 2006.

[5] M. T. Stahl, "Open-source software: not quite endsville," *Drug Discovery Today*, vol. 10, no. 3, pp. 219–222, 2005.

[6] W. L. DeLano, "The case for open-source software in drug discovery," *Drug Discovery Today*, vol. 10, no. 3, pp. 213–217, 2005.

[7] P. B. de Laat, "Copyright or copyleft? An analysis of property regimes for software development," *Research Policy*, vol. 34, no. 10, pp. 1511–1532, 2005.

[8] A. Fuggetta, "Open source software—an evaluation," *Journal of Systems and Software*, vol. 66, no. 1, pp. 77–90, 2003.

[9] A. Gambardella and B. H. Hall, "Proprietary versus public domain licensing of software and research products," *Research Policy*, vol. 35, no. 6, pp. 875–892, 2006.

[10] S. Krishnamurthy, "A managerial overview of open source software," *Business Horizons*, vol. 46, no. 5, pp. 47–56, 2003.

[11] K. R. Lakhani and E. von Hippel, "How open source software works: "free" user-to-user assistance," *Research Policy*, vol. 32, no. 6, pp. 923–943, 2003.

[12] E. S. Raymond, "The Cathedral and the Bazaar," 1997, http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar.

[13] W. L. Oberkampf and T. G. Trucano, "Verification and validation in computational fluid dynamics," Sandia Report SAND2002-0529, Sandia National Laboratories, Albuquerque, NM, USA, March 2002.

[14] A. Srivastava and S. Sundaresan, "Analysis of a frictional-kinetic model for gas-particle flow," *Powder Technology*, vol. 129, no. 1–3, pp. 72–85, 2003.

[15] D. Boyalakuntla, Personal communication, 2006.

[16] S. Zaleski, "Science and Fluid Dynamics should have more open sources," October 2001, http://www.lmm.jussieu.fr/~zaleski/OpenCFD.html.

[17] M. Syamlal, W. Rogers, and T. J. O'Brien, "MFIX Documentation: Theory Guide," Technical Note DOE/METC-94/1004, NTIS/DE94000087, National Technical Information Service, Springfield, Va, USA, 1993.

[18] M. Syamlal, "MFIX Documentation, User's Manual," Technical Note DOE/METC-95/1013, NTIS/DE95000031, National Technical Information Service, Springfield, Va, USA, 1994.

[19] M. Syamlal, S. Venkatesan, and S. M. Cho, "Modeling of coal conversion in a carbonizer," in *Proceedings of 13th Annual International Pittsburgh Coal Conference*, S.-H. Ciang, Ed., vol. 2, pp. 1309–1314, University of Pittsburgh, Pittsburgh, Pa, USA, September 1996.

[20] C. Guenther, M. Shahnam, M. Syamlal, J. Longanbach, D. Cicero, and P. Smith, "CFD modeling of a transport gasifier," in *Proceedings of the 19th Annual Pittsburgh Coal Conference*, Pittsburgh, Pa, USA, September 2002.

[21] D. Gidaspow, *Multiphase Flow and Fluidization: Continuum and Kinetic Theory Descriptions*, Academic Press, New York, NY, USA, 1994.

[22] S. Benyahia, M. Syamlal, and T. J. O'Brien, "Summary of MFIX Equations 2005-4," March 2006, http://www.mfix.org/documentation/MfixEquations2005-4-1.pdf.

[23] M. Syamlal, "MFIX Documentation: Numerical Techniques," DOE/MC-31346-5824. NTIS/DE98002029, December 1998.

[24] E. D'Azevedo, S. Pannala, M. Syamlal, A. Gel, M. Prinkey, and T. J. O'Brien, "Parallelization of MFIX: a multiphase CFD code for modeling fluidized beds," in *Proceedings of the 10th SIAM Conference on Parallel Processing for Scientific Computing*, Portsmouth, Va, USA, March 2001.

[25] S. Pannala, E. D'Azevedo, T. J. O'Brien, and M. Syamlal, "Hybrid (mixed SMP/DMP) parallelization of MFIX: a multiphase CFD code for modeling fluidized beds," in *Proceedings of ACM Symposium on Applied Computing*, Melbourne, Fla, USA, March 2003.

[26] A. Gel, S. Pannala, M. Syamlal, T. J. O'Brien, and E. S. Gel, "Comparison of frameworks for a next-generation multiphase flow solver, MFIX: a group decision-making exercise," *Concurrency and Computation: Practice and Experience*, vol. 19, no. 5, pp. 609–624, 2006.

[27] S. J. Gelderbloom, D. Gidaspow, and R. W. Lyczkowski, "CFD simulations of bubbling/collapsing fluidized beds for three Geldart groups," *AIChE Journal*, vol. 49, no. 4, pp. 844–858, 2003.

[28] J. De Wilde, "Reformulating and quantifying the generalized added mass in filtered gas-solid flow models," *Physics of Fluids*, vol. 17, no. 11, Article ID 113304, 14 pages, 2005.

[29] X. Wang, C. Zhu, and R. Ahluwalia, "Numerical simulation of evaporating spray jets in concurrent gas-solids pipe flows," *Powder Technology*, vol. 140, no. 1-2, pp. 56–67, 2004.

[30] C. R. Sutton and J. C. Chen, "Dynamic behavior of local solids concentration in fluidized beds: experimental validation of an Eulerian-Eulerian model," in *Proceedings of the AIChE Annual Meeting*, Austin, Tex, USA, November 2004.

[31] J. Sun and F. Battaglia, "Hydrodynamic modeling of particle rotation for segregation in bubbling gas-fluidized beds," *Chemical Engineering Science*, vol. 61, no. 5, pp. 1470–1479, 2006.

[32] J. Sun and F. Battaglia, "Effects of particle rotation on the hydrodynamics modeling of segregation in gas-fluidized beds," in *Proceedings of the ASME Fluids Engineering Division (IMECE '04)*, vol. 260, pp. 745–753, Anaheim, Calif, USA, March 2004.

[33] S. Sundaresan, "Perspective: modeling the hydrodynamics of multiphase flow reactors: current status and challenges," *AIChE Journal*, vol. 46, no. 6, pp. 1102–1105, 2000.

[34] K. Agrawal, P. N. Loezos, M. Syamlal, and S. Sundaresan, "The role of meso-scale structures in rapid gas-solid flows," *Journal of Fluid Mechanics*, vol. 445, pp. 151–185, 2001.

[35] P. N. Loezos and S. Sundaresan, "The role of meso-scale structures on dispersion in gas-particle flows," in *Circulating Fluidized Beds VII*, J. R. Grace, J. Zhu, and H. I. de Lasa, Eds., pp. 427–434, Canadian Society for Chemical Engineering, Ottawa, Ontario, Canada, 2002.

[36] A. T. Andrews, P. N. Loezos, and S. Sundaresan, "Coarse grid simulation of gas particle flows in vertical risers," *Industrial & Engineering Chemistry Research*, vol. 44, no. 16, pp. 6022–6037, 2005.

[37] M. W. Weber and C. M. Hrenya, "Square-well model for cohesion in fluidized beds," *Chemical Engineering Science*, vol. 61, no. 14, pp. 4511–4527, 2006.

[38] C. M. Hrenya, University of Colorado, Personal communication, 2005.

[39] R. Fan, D. L. Marchisio, and R. O. Fox, "DQMOM model for gas-solid fluidized bed with aggregation and breakage," in *Computational Fluid Dynamics in Chemical Engineering III*, Davos, Switzerland, May 2003.

[40] R. Fan, D. L. Marchisio, and R. O. Fox, "Application of the direct quadrature method of moments to poly-disperse, gas-solid fluidized beds," *Powder Technology*, vol. 139, no. 1, pp. 7–20, 2004.

[41] N. Xie, F. Battaglia, and R. O. Fox, "Simulations of multiphase reactive flows in fluidized beds using *in situ* adaptive tabulation," *Combustion, Theory, and Modelling*, vol. 8, no. 2, pp. 195–209, 2004.

[42] N. Xie, "Simulations of multiphase reactive flows in fluidized beds using *in situ* adaptive tabulation method," M.S. thesis, Iowa State University, Ames, Iowa, USA, August 2002.

[43] N. Xie, F. Battaglia, K. J. Timmer, and R. C. Brown, "Modeling of elutriation phenomenon in fluidized beds," in *Proceedings of the 58th Annual Meeting of the Division of Fluid Dynamics*, American Physical Society, Chicago, Ill, USA, November 2005.

[44] Y. T. Makkawi and R. Ocone, "Validation of CFD model for fluidised bed over broad ranges of operating conditions," in *Proceedings of the 5th World Congress on Particles Technology*, Orlando, Fla, USA, April 2006.

[45] Y. T. Makkawi, P. C. Wright, and R. Ocone, "The effect of friction and inter-particle cohesive forces on the hydrodynamics of gas-solid flow: a comparative analysis of theoretical predictions and experiments," *Powder Technology*, vol. 163, no. 1-2, pp. 69–79, 2006.

[46] T. McKeen and T. Pugsley, "Simulation of cold flow FCC stripper hydrodynamics at small scale using computational fluid dynamics," *International Journal of Chemical Reactor Engineering*, vol. 1, article A18, 2003.

[47] T. McKeen and T. Pugsley, "Simulation and experimental validation of freely bubbling bed of FCC catalyst," *Powder Technology*, vol. 129, no. 1–3, pp. 139–152, 2003.

[48] S. D. Sharma, T. Pugsley, and R. Delatour, "Three-dimensional CFD model of the deaeration rate of FCC particles," *AIChE Journal*, vol. 52, no. 7, pp. 2391–2400, 2006.

[49] R. Fan, R. O. Fox, and M. E. Muhle, "CFD validation of a polyethylene pilot-scale fluidized bed," in *Proceedings of the 3rd European Conference on the Reaction Engineering of Polyolefins*, Lyon, France, June 2005.

[50] G. Bergantz, University of Washington, Personal communication, 2005.

[51] L. Oger, University of Rennes and McGill University, Personal communication, 2005.

[52] S. Dartevelle and G. A. Valentine, "Early-time multiphase interactions between basaltic magma and underground repository openings at the proposed Yucca Mountain radioactive waste repository," *Geophysical Research Letters*, vol. 32, L22311, 2005.

[53] S. Dartevelle, "Comprehensive Approaches to Multiphase Flows in Geophysics. Application to non isothermal, non homogenous, unsteady, large scale, turbulent dusty clouds. I. Basic RANS and LES Navier Stokes equations," LA 14228, pp.51, Los Alamos National Laboratory, Los Alamos, New Mexico, 2005.

[54] S. Dartevelle, "Numerical modeling of geophysical granular flows: 1. A comprehensive approach to granular rheologies and geophysical multiphase flows," *Geochemistry, Geophysics, Geosystems*, vol. 5, no. 8, Q08003, 2004.

[55] S. Dartevelle, W. I. Rose, J. Stix, K. Kelfoun, and J. W. Vallance, "Numerical modeling of geophysical granular flows: 2. Computer simulations of plinian clouds and pyroclastic flows and surges," *Geochemistry, Geophysics, Geosystems*, vol. 5, no. 8, Q08004, 2004.

[56] N. Siegel, Sandia National Laboratory, Personal communication, 2005.

[57] U. Imke, Forschungszentrum Karlsruhe GmbH, Personal communication, 2005.

[58] S. Dartevelle, "Numerical and granulometric approaches to geophysical granular flows," Ph.D. dissertation, Department of Geological and Mining Engineering, Michigan Technological University, Houghton, Mich, USA, 2003.

[59] K. Agrawal, "The role of meso-scale structures in rapid granular and gas solid flows," Ph.D. thesis, Princeton University, Princeton, NJ, USA, 2000.

[60] A. Srivastava, "Dense phase gas solid flows in circulating fluidized beds," Ph.D. thesis, Princeton University, Princeton, NJ, USA, 2002.

[61] P. N. Loezos, "An investigation into dense and dilute gas particle flow," Ph.D. thesis, Princeton University, Princeton, NJ, USA, 2003.