Extending LMS to Support IRT-Based Assessment Test Calibration

Panagiotis Fotaris, Theodoros Mastoras, Ioannis Mavridis, and Athanasios Manitsaris

Department of Applied Informatics, University of Macedonia,
156 Egnatia str., 54006 Thessaloniki, Greece
{paf,mastoras,mavridis,manits}@uom.gr

Abstract. Developing unambiguous and challenging assessment material for measuring educational attainment is a time-consuming, labor-intensive process. As a result Computer Aided Assessment (CAA) tools are becoming widely adopted in academic environments in an effort to improve the assessment quality and deliver reliable results of examinee performance. This paper introduces a methodological and architectural framework which embeds a CAA tool in a Learning Management System (LMS) so as to assist test developers in refining items to constitute assessment tests. An Item Response Theory (IRT) based analysis is applied to a dynamic assessment profile provided by the LMS. Test developers define a set of validity rules for the statistical indices given by the IRT analysis. By applying those rules, the LMS can detect items with various discrepancies which are then flagged for review of their content. Repeatedly executing the aforementioned procedure can improve the overall efficiency of the testing process.

Keywords: e-learning, Assessment Test Calibration, Computer Aided Assessment, Item Analysis, Item Response Theory, Learning Management System.

1 Introduction

With the proliferation of computer and Internet technologies, *Computer Aided Assessment* (*CAA*) tools have become a major trend in academic institutions worldwide. Through these systems, tests composed of various question types can be presented to students in order to assess their knowledge. Yet, there has been considerable criticism of the test quality, with both research and experience showing that many test items (questions) are flawed in some way at the initial stage of their development. Test developers can expect about 50% of their items will fail to perform as intended which may eventually lead to unreliable results of examinee performance [1]. It is therefore imperative to assure that the individual test items are of the highest quality possible since a poor one could have an inordinately large effect on some scores.

There are two major approaches to item evaluation using item response data, and both can be used, sample size permitting. The classical approach focuses on traditional item indices borrowed from *Classical Test Theory* (*CTT*) such as item difficulty, item discrimination, and the distribution of examinee responses across the alternative

responses. The second approach uses *Item Response Theory* (*IRT*) to estimate the parameters of an item characteristic curve which provides the probability that an item will be answered correctly based on the examinee's ability level as measured by the test.

The natural scale for item difficulty in CTT is the percentage of examinees correctly answering the item. One term of item difficulty is *p-value*, which stands for the proportion of percentage of examinees correctly answering the item. Every item has a natural difficulty based on the performance of all persons undertaking the test; however, this p-value is quite difficult to estimate accurately unless a very representative group of test-takers is being tested. If for example the sample contains well instructed, highly able or highly trained people, then the test and its items will appear very easy. On the other hand, if the sample contains uninstructed, low-ability or untrained people, then the same test will appear very hard. This is one of the main reasons that CTT is often criticized for [2], [3], because the estimation of the p-value is potentially biased by the sample on which the estimate of item difficulty is based.

With IRT the composition of the sample is generally immaterial, and item difficulty can be estimated without bias. The one-, two-, and three-parameter binary-scoring (dichotomous) IRT models typically lead to similar estimates of difficulty, and these estimates are highly correlated to classical estimates of difficulty. Additionally, while classical statistics are relatively simple to compute and understand and do not require sample sizes as large as those required by IRT statistics, they a) are not as likely to be as sensitive to items that discriminate differentially across different levels of ability (or achievement), b) do not work as well when different examinees take different sets of items, and c) are not as effective in identifying items that are statistically biased [4]. As a result, the use of IRT models spread rapidly during the last 20 years and they are now used in the majority of large-scale educational testing programs involving 500 or most test-takers.

IRT analysis yields three estimated parameters for each item, α, b and c respectively. The α parameter is a measure of the discriminating power of the item, the b parameter is an index of item difficulty, and the c is the "guessing" parameter, defined as the probability of a very low-ability test taker getting the item correct. A satisfactory pool of items for testing is one characterized by items with high discrimination ($\alpha > 1$), a rectangular distribution of difficulty (b), and low guessing ($c < 0.2$) parameters [5], [6]. The information provided by the item analysis assists not only in evaluating performance but in improving item quality as well. Test developers can use these results to discriminate whether an item can be reused as is, should be revised before reuse or should be taken out of the active item pool. What makes an item's performance acceptable should be defined in the test specifications within the context of the test purpose and use.

Unfortunately only a few test developers have the statistical background needed to fully understand and utilize the IRT analysis results. Although it is almost impossible to compel them to further their studies, it is possible to provide them with some feedback regarding the quality of the test items. This feedback can then act as a guide to discard defective items or to modify them in order to improve their quality for future use. Based on that notion, the present paper introduces a comprehensible way to

present IRT analysis results to test developers without delving into unnecessary details. Instead of memorizing numerous commands and scenarios from technical manuals, test developers can easily detect problematic questions from the familiar user interface of a *Learning Management System* (*LMS*). The latter can automatically calculate the limits and rules for the α, b, and c parameters based on the percentage of questions wanted for revision. The examinee's *proficiency* (θ) is represented on the usual scale (or metric) with values ranging roughly between -3 and 3, but since these scores include negative ability estimates which would undoubtedly confuse many users, they can optionally be normalized to a 0..100 range scale score.

2 Related Works

The use of *Learning Management Systems* (*LMSs*) and CAA tools has increased greatly due to the students' demand for more flexible learning options. However, only a small fraction of these systems supports an assessment quality control process based on the interpretation of item statistic parameters. Popular e-learning platforms such as Blackboard [7], Moodle [8] and Questionmark [9] have plug-ins or separated modules that provide statistics for test items, but apart from that they offer no suggestions to test developers on how to improve the problematic items. Therefore, many researchers have recently endeavored to provide mechanisms for test calibration.

Hsieh et al. introduced a model that presents test statistics and collects students' learning behaviors for generating analysis result and feedback to tutors [10]. Hung et al. proposed an analysis model based on CTT that collects information such as item difficulty and discrimination indices, questionnaire and question style etc. These data are combined with a set of rules in order to detect defective items, which are signaled using traffic lights [11]. Costagliola et al.'s eWorkbook system improved that idea by using fuzzy rules to measure item quality, detect anomalies on the items, and give advice for their improvement [12]. Nevertheless, all of the aforementioned works preferred CTT to IRT for ease of use without taking into consideration its numerous deficiencies.

On the other hand, IRT has been mainly applied in the *Computerized Adaptive Test* (*CAT*) domain for personalized test construction based on individual ability [13], [14], [15], [16], [17]. Despite its high degree of support among theoreticians and some practitioners, IRT's complexity and dependence on unidimensional test data and large samples often relegate its application only to experimental purposes. While a literature review can reveal many different IRT estimation algorithms, they all involve heavy mathematics and are unsuitable for implementation in a scripting language designed for web development (i.e. PHP). As a result, their integration in internet applications such as LMSs is very limited. A way to address this issue is to have a webpage call the open-source analysis tool ICL to carry out the estimation process and then import its results for display. The present paper showcases a framework that follows the aforementioned method in order to extend an LMS with IRT analysis services at no extra programming cost.

3 Open-Source IRT Analysis Tool ICL

Several computer programs that provide estimates of IRT parameters are currently available for a variety of computer environments [18], [19]. These include Rascal [20], Ascal [21], WINSTEPS [22], BILOG-MG [23], MULTILOG [24], PARSCALE [25], [26], RUMM [27] and WINMIRA [28] to name a few that are easily obtainable. Despite being the de facto standard for dichotomous IRT model estimation, BILOG is a commercial product and limited in other ways. Hanson provided an alternative stand-alone software for estimating the parameters of IRT models called *IRT Command Language* (*ICL*) [29]. A recent comparison between BILOG-MG and ICL [30] showed that both programs are equally precise and reliable in their estimations. However, ICL is a free, open-source licensed in a way that allows it to be modified and extended. In fact, ICL is actually IRT estimation functions (ETIRM) [31] embedded into a fully-featured programming language called *TCL* ("tickle") [32] and thus allowing relatively complex operations. Additionally, ICL's command line nature enables it to run in the background and produce analysis results in the form of text files. Since the proposed framework uses only a three-parameter binary-scoring IRT model (*3PL*), ICL proves more than sufficient for our purpose and was therefore selected to complement the LMS for assessment test calibration.

4 Integrating IRT Analysis in Dokeos LMS

Dokeos is an open-source LMS accompanied by Free Software Foundation's [33] [34] General Public License [35]. It is implemented in PHP and requires Apache acting as a web server and mySQL as a Database Management System. Dokeos has been serving the needs of two academic courses at the University of Macedonia for over four years, receiving satisfactory feedback from both instructors and students. In order to extend its functionality with IRT analysis and assessment test calibration functions, we had to modify the source code so as to support the following features:

1. After completing a test session, the LMS stores in its database the examinee's response to each test item instead of keeping only a final score by default.
2. Test developers define the acceptable limits for the following IRT analysis parameters: a) item discrimination, b) item difficulty, and c) guessing. The LMS stores these values as validity rules for each assessment. There is an additional choice of having these limits set automatically by the system in order to rule out a specific percentage of questions (Fig. 1.1).
3. Every time the LMS is asked to perform an IRT analysis, it displays a page with the estimated difficulty, discrimination and guessing parameters for each assessment item. If the latter violates any of the validity rules already defined in the assessment profile, it is flagged for review of its content (Fig. 1.2). Once item responses are evaluated, test developers can discard, revise or retain items for future use.
4. In addition to a total score, the assessment report screen displays the proficiency θ per examinee as derived from the IRT analysis (Fig. 1.3).

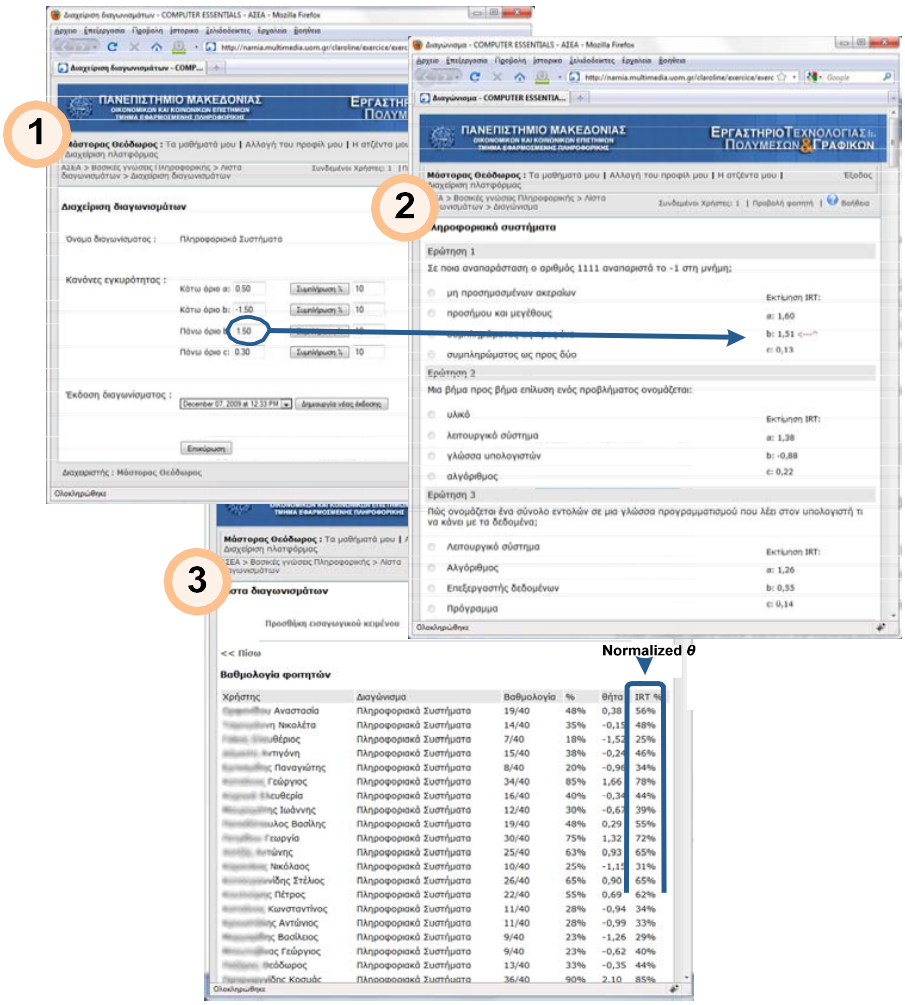

Fig. 1. Functionality features supported in extended Dokeos LMS

5 The Proposed Item Analysis Methodology

The proposed methodology consists of four steps, with each one of them being an action performed by the LMS. Although we used Dokeos as our LMS of choice, the proposed item analysis methodology can be applied to other e-learning tools, too. Once an update of the IRT results is called for, the LMS exports the proper data files and TCL scripts (Fig. 3). The LMS then performs a number of calls to the ICL using PHP (Fig. 4 and 5) and after parsing the analysis results, it imports them to its database. A system following this approach is illustrated in Fig. 2.

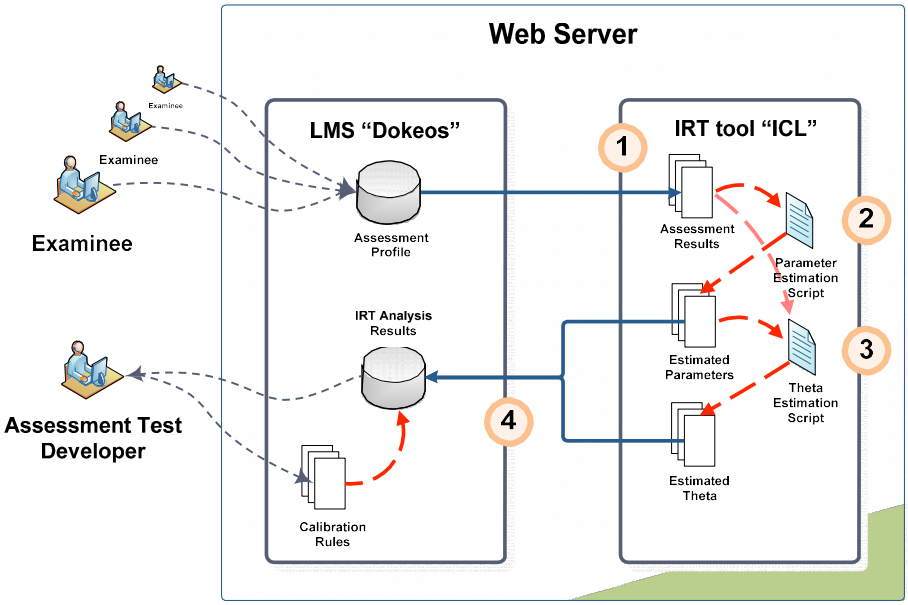

Fig. 2. System architecture

The proposed methodology consists of the following steps:

1. The LMS exports the assessment results to a data file and generates a TCL script to process them (parameter estimation script). The bold parts in the script change after each execution, depending on the number of the test items and the assessment name (e.g. *40* and *test0140* respectively). The rest of the script is about the algorithm performed by the ICL ("EM" algorithm), the type of IRT analysis (dichotomous) and the maximum number of iterations (200).
2. The LMS then calls up ICL with the parameter estimation script passed as a parameter in order to create a data file containing the α, b, and c values for each test item. At the same time it prepares a second TCL script to process these IRT parameters (θ estimation script).
3. The LMS calls up ICL with the θ estimation script passed as a parameter so as to make a data file with the examinees' θ values.
4. Finally, the LMS imports the two ICL-produced data files (*.par and *.theta) to its database for further processing in the context of the aimed assessment test calibration.

Once an initial item pool has been calibrated, examinees can then be tested routinely. As time goes on, it would almost surely become desirable to retire items that are flawed, have become obsolete, or have been used many times, and to replace them with new items. Having these problematic items already been detected by the LMS, test developers can take any necessary course of action to improve the quality of tests. Additionally, since the limits for the IRT analysis parameters are not hard-coded, test developers can modify them at will in order to tune the sensitivity of the system.

```
0101001000100111111001010101100000100111
0101000100010001100000111001100000100110
0000000000000011000000110001000010001000
0001010000110010100000111101110010000100
0100010000000001100000000001001010000100
0111011101110111111101111111111101111111
1111001001110000000000011101010000101100
0110000000010011101000110000001000000110
.
.
.
…… one row per examinee ……
```

```
output -no_print
allocate_items_dist 40
read_examinees test0140.dat 40i1
starting_values_dichotomous
EM_steps -max_iter 200
print -item_param
release_items_dist
```

Fig. 3. (a) Assessment results (test0140.dat file). (b) Parameter Estimation Script (test0140.tcl file).

```
1   1,597597   1,506728   0,128515
2   1,377810   -0,876164  0,223903
3   1,258461   0,549362   0,140593
4   1,031856   0,495642   0,079279
5   1,077831   1,004437   0,136324
6   0,479151   1,544218   0,218270
7   1,439241   1,279352   0,082382
8   0,898259   1,310215   0,129570
9   1,837514   1,349520   0,032675
10  0,467694   0,934207   0,206085
11  0,607603   0,265524   0,181212
12  0,240009   1,054301   0,245737
13  0,945631   1,451464   0,050895
.
.
.
…… one row per item ……
```

```
output -no_print
allocate_items_dist 40
read_examinees test0140.dat 40i1
read_item_param test0140.par
set estep [new_estep]
estep_compute $estep 1 1
delete_estep $estep
set eapfile [open test0140.theta w]
for {set i 1} {$i <=
[num_examinees]} {incr i} {
.
.
.
}
close $eapfile
release items dist
```

Fig. 4. (a) Estimated parameters (test0140.par file). (b) θ estimation script (test0140t.tcl file).

```
0,378453     0,434304       19
-0,149162    -0,096175      14
-1,523733    -5,999491      7
-0,238032    -0,172708      15
-0,964941    -1,001566      8
1,658672     1,737581       34
-0,343387    -0,312642      16
-0,665486    -0,666954      12
.
.
.
…… one row per examinee ……
```

Fig. 5. Estimated theta (test0140.theta file)

6 Conclusion

The present paper introduced a methodological and architectural framework for extending an LMS with IRT–based assessment test calibration. Instead of having web developers implement complex IRT estimation algorithms within the LMS, the proposed methodology uses ICL to obtain reliable IRT analysis results. The latter are then automatically imported to the LMS, thus releasing test developers of this burdensome duty. By applying a set of validity rules, the enhanced LMS is able to detect several defective items which are then reported for review of their content. As a result, the suggested approach is capable of assisting test developers in their continuous effort to improve flaws test items. Moreover, the user-friendly interface allows users with no previous expertise in statistics to comprehend and utilize the IRT analysis results.

According to research focused on IRT sample size effects [36], a great number of examinees are needed to obtain accurate results. For example, Swaminathan and Gifford [37] concluded that about 1,000 examinees are required when using the 3PL model. This would pose a problem for most test developers due to the fact that the number of examinees in academic courses rarely exceeds 150. Nevertheless, less accurate estimates are acceptable when aiming for assessment calibration since the desired goal is to identify test items with the highest and lowest parameter values. The proposed system introduces a feature that addresses the aforementioned issue (Fig. 1.1) and allows test developers to easily pinpoint this particular group of test items for revision.

This initial experiment produced encouraging results, showing that the system can effectively evaluate item performance and therefore increase the overall validity of the testing process. The fact that the proposed methodology is not limited to Dokeos but can be easily adopted by different e-learning environments makes it especially suitable for academic use.

References

1. Haladyna, T.M.: Developing and Validating Multiple-Choice Test Items, 2nd edn. Lawrence Erlbaum Associates, Mahwah (1999)
2. Hambleton, R.K., Jones, R.W.: Comparison of Classical Test Theory and Item Response Theory and their Applications to Test Development. Educational Measurement: Issues and Practices 12, 38–46 (1993)
3. Hambleton, R.K., Swaminathan, H.: Item Response Theory: Principles and Applications. Kluwer-Nijhoff Publishing, Boston (1987)
4. Schmeiser, C.B., Welch, C.J.: Test Development. In: Brennan, R.L. (ed.) Educational Measurement, 4th edn. Praeger Publishers, Westport (2006)
5. Flaugher, R.: Item Pools. In: Wainer, H. (ed.) Computerized Adaptive Testing: A Primer, 2nd edn. Lawrence Erlbaum Associates, Mahwah (2000)
6. Baker, F.B.: Item Response Theory: Parameter Estimation Techniques. Marcel Dekker, New York (1992)
7. Moodle.org: Open-source Community-based Tools for Learning, `http://moodle.org/`
8. Blackboard Home, `http://www.blackboard.com`

9. Questionmark...Getting Results, http://www.questionmark.com
10. Hsieh, C., Shih, T.K., Chang, W., Ko, W.: Feedback and Analysis from Assessment Metadata in E-learning. In: 17th International Conference on Advanced Information Networking and Applications (AINA 2003), pp. 155–158. IEEE Computer Society, Los Alamitos (2003)
11. Hung, J.C., Lin, L.J., Chang, W., Shih, T.K., Hsu, H., Chang, H.B., Chang, H.P., Huang, K.: A Cognition Assessment Authoring System for E-Learning. In: 24th International Conference on Distributed Computing Systems Workshops (ICDCS 2004 Workshops), pp. 262–267. IEEE Computer Society, Los Alamitos (2004)
12. Costagliola, G., Ferrucci, F., Fuccella, V.: A Web-Based E-Testing System Supporting Test Quality Improvement. In: Leung, H., Li, F., Lau, R., Li, Q. (eds.) ICWL 2007. LNCS, vol. 4823, pp. 264–275. Springer, Heidelberg (2008)
13. Wu, I.L.: Model management system for IRT-based test construction decision support system. Decision Support Systems 27(4), 443–458 (2000)
14. Chen, C.M., Duh, L.J., Liu, C.Y.: A Personalized Courseware Recommendation System Based on Fuzzy Item Response Theory. In: IEEE International Conference on e-Technology, e-Commerce and e-Service, pp. 305–308. IEEE Computer Society Press, Los Alamitos (2004)
15. Ho, R.G., Yen, Y.C.: Design and Evaluation of an XML-Based Platform-Independent Computerized Adaptive Testing System. IEEE Transactions on Education 48(2), 230–237 (2005)
16. Sun, K.: An Effective Item Selection Method for Educational Measurement. In: Advanced Learning Technologies, pp. 105–106 (2000)
17. Yen, W., Fitzpatrick, A.R.: Item Response Theory. In: Brennan, R.L. (ed.) Educational Measurement, 4th edn. Praeger Publishers, Westport (2006)
18. Kim, S., Cohen, A.S.: A Comparison of Linking and Concurrent Calibration under Item Response Theory. Applied Psychological Measurement 22(2), 131–143 (1998)
19. Embretson, S.E., Reise, S.P.: Item Response Theory for Psychologists. Lawrence Erlbaum, Mahwah (2000)
20. Assessment System Corporation: RASCAL (Rasch Analysis Program). Computer Software, Assessment Systems Corporation, St. Paul, Minnesota (1992)
21. Assessment System Corporation: ASCAL (2- and 3-parameter) IRT Calibration Program. Computer Software, Assessment Systems Corporation, St. Paul, Minnesota (1989)
22. Linacre, J.M., Wright, B.D.: A user's guide to WINSTEPS. MESA Press, Chicago (2000)
23. Zimowski, M.F., Muraki, E., Mislevy, R.J., Bock, R.D.: BILOG-MG 3: Multiple-group IRT analysis and test maintenance for binary items. Computer Software, Scientific Software International, Chicago (1997)
24. Thissen, D.: MULTILOG user's guide. Computer Software, Scientific Software International, Chicago (1991)
25. Muraki, E., Bock, R.D.: PARSCALE: IRT-based Test Scoring and Item Analysis for Graded Open-ended Exercises and Performance Tasks. Computer Software, Scientific Software International, Chicago (1993)
26. du Toit, M. (ed.): IRT from SSI. Scientific Software International. Lincolnwood, Illinois (2003)
27. Andrich, D., Sheridan, B., Luo, G.: RUMM: Rasch Unidimensional Measurement Model. Computer Software, RUMM Laboratory, Perth, Australia (2001)
28. von Davier, M.: WINMIRA: Latent Class Analysis, Dichotomous and Polytomous Rasch Models. Computer Software, Assessment Systems Corporation, St. Paul, Minnesota (2001)

29. Hanson, B.A.: IRT Command Language (ICL). Computer Software, http://www.b-a-h.com/software/irt/icl/index.html
30. Mead, A.D., Morris, S.B., Blitz, D.L.: Open-source IRT: A Comparison of BILOG-MG and ICL Features and Item Parameter Recovery, http://mypages.iit.edu/~mead/MeadMorrisBlitz2007.pdf
31. Hanson, B.A.: Estimation Toolkit for Item Response Models (ETIRM). Computer Software, http://www.b-a-h.com/software/cpp/etirm.html
32. Welch, B.B., Jones, K., Hobbs, J.: Practical programming in Tcl and Tk, 4th edn. Prentice Hall, Upper Saddle River (2003)
33. Open Source Initiative, http://www.opensource.org/docs/definition.php
34. General Public License, http://www.gnu.org/copyleft/gpl.html
35. Free Software Foundation, http://www.fsf.org/
36. Hulin, C.L., Lissak, R.I., Drasgow, F.: Recovery of Two- and Three-parameter Logistic Item Characteristic Curves: A Monte Carlo Study. Applied Psychological Measurement 6(3), 249–260 (1982)
37. Swaminathan, H., Gifford, J.A.: Estimation of Parameters in the Three-parameter Latent Trait Model. In: Weiss, D.J. (ed.) New Horizons in Testing. Academic Press, New York (1983)