Chapter 13
Pedagogical and Organisational Issues in the Campaign for IT Literacy Through Spoken Tutorials

Kannan M. Moudgalya

Abstract This chapter describes an IT literacy promotion drive in India, using Spoken Tutorials, offered through organised self-learning workshops. Spoken Tutorials are Screencast videos of 10-min duration. The steps taken to make the Spoken Tutorials suitable for self-learning are described. As the Spoken Tutorial activity is restricted to open-source software, practice-based learning is possible, thereby being as effective as active learning. Only the spoken part of these tutorials is dubbed into Indian languages, benefiting a large number of students who are weak in English, which is nevertheless widely spoken in India. This helps everyone learn important IT topics while retaining the employment potential. About 100,000 students are trained every year through this methodology, with an option to earn certificates on passing online tests, all of which are offered absolutely free of cost. The website http://spoken-tutorial.org, which is used for coordination, has become extremely popular, with the number of page views expected to be three to four million in the current year. Feedback from 25,000 students and testimonials vouch for the usefulness of the Spoken Tutorial methodology.

Abbreviations

FOSS:	Free and Open Source Software
IGNOU:	Indira Gandhi National Open University
IT:	Information Technology
SELF Workshop:	Spoken tutorial based Education and Learning through Free FOSS study Workshop
ST:	Spoken Tutorial
STs:	Spoken Tutorials

K.M. Moudgalya (✉)
Department of Chemical Engineering, Education Technology, Systems & Control,
Indian Institute of Technology Bombay, Powai, Mumbai 400 076, India
e-mail: kannan@iitb.ac.in

R. Huang et al. (eds.), *The New Development of Technology Enhanced Learning: Concept, Research and Best Practices*, Lecture Notes in Educational Technology,
DOI 10.1007/978-3-642-38291-8_13, © Springer-Verlag Berlin Heidelberg 2014

13.1 Introduction

In this work, we give an account of the steps taken to spread IT literacy in India through Spoken Tutorials (STs) (Moudgalya 2009, 2011a, 2013; Spoken Tutorial Project). This work has been undertaken to improve the employment potential in India. We have the following reasons to undertake this activity: The education sector in India is undergoing a crisis. While there are attempts to increase the Gross Enrollment Ratio, the number of people who can be employed is abysmally low, according to a McKinsey report (Farrell et al. 2005). One of the reasons for this deplorable situation is the shortage of good quality teachers (Moudgalya et al. 2008). On the other hand, in spite of the economic slowdown, industry experts report a large shortage of trained people, especially in the IT sector (Geron 2013). The Spoken Tutorial (ST) project is funded by the Ministry of Human Resource Development of the Government of India to improve the IT literacy of Indian students, so as to improve their employment potential.

In areas of activity-based learning, there is sufficient reason to believe that computer-aided instructions may be as effective as traditional, in-person instructions (Koufogiannakis and Wiebe 2006). This makes a compelling case to use computer-aided instructions when the quality of teachers is wanting. Screencast-based tutorials have effectively been used in various fields (Mohorovičić 2012). Moseley (2013) explains the many benefits of using the instructional material created using Screencast.

An ST is an audio-video instructional material *created for self-learning* through the Screencast technology. The objectives of the ST effort are:

1. To create documentation for Free and Open Source Software (FOSS). Restriction to FOSS promotes active learning, along with other benefits. In a typical Spoken Tutorial of 10-min duration, there are about 100 screen transitions. It is a lot more difficult to create an equally effective PDF document using screenshots. Generally, the Screencasts are known to be more effective to understand a new topic (Novaković et al. 2013).
2. To make every ST suitable for self-learning, given that there is a big shortage of good teachers in India.
3. To conduct workshops using ST, so as to make it easy for students to access the instructional material.
4. To make IT literacy training accessible to students weak in English, without affecting their employment potential.
5. To come up with a mechanism to conduct tests for the participants of ST-based workshops and to issue certificates to the passing students.

All of the above should be made available free of cost to the learners. We will now discuss how we went about to achieve these objectives in the above list while highlighting the pedagogical issues.

13.2 Content Creation

Standard guidelines to create good Screencasts are available (ST Team; Moseley 2013). Creating content suitable for self-learning, when the target audience is large and disparate, is a more difficult task, requiring meticulous planning and adhering to strict guidelines. In this section, we discuss the issues in the creation of outline, script, recording, translation and dubbing in the ST methodology. We also discuss the process of novice check that we have introduced to achieve our objective.

13.2.1 Outline

STs should be classified in a convenient way, for example, as beginner, intermediate and expert. We insist on an outline before one embarks on creation of STs. Outline for each tutorial should be presented in approximately ten bullet points. We recommend an *overview tutorial* for each classification, in general, and the beginner level, in particular. We recommend that the overview tutorial be created after creating the constituent tutorials, although the learner will watch them in the reverse order. This helps ensure that the overview tutorial also is activity based, as explained in Sect. 13.2.2.

The target content should be chunked into STs of 10-min duration, chosen to address the issue of short attention span. The reason why it is not made shorter is to minimise the fraction of overheads in each tutorial, such as information about our project, whom to contact to organise workshops, etc., which are useful to a person who happens to watch just one tutorial. Although Guo (2013) recommends an optimal video length to be of 6 min, the attention seems to be about the same up to 9 min in their study. Restricting the length to 10 min forces the creator to be focused and to cut out the unnecessary information. A procedure to estimate the length of a tutorial is given in Sect. 13.2.2.

The STs should be organised so as to avoid repetition and to ensure smooth flow without having to go back and forth. We insist on a few examples to run through the tutorials. We encourage the files created in a tutorial, if any, to be used in subsequent tutorial and be made available to the users through our web interface, as explained in Sect. 13.3.7.

13.2.2 Script

We require a script to be written before creating a tutorial as it provides many advantages: If a section of the tutorial has to be improved, one can rewrite the script and record it. On the basis of a script, a novice can decide whether they can reproduce all the described actions. It is possible to stream this script when a tutorial is played back, helping students weak in English (Mitterer and McQueen 2009) and the hearing impaired. It is useful to search for key terms. If timing of the script is available, one may quickly locate the required part of a video during the recall stage, thereby addressing one of the shortcomings of Screencasts (Novaković et al. 2013). Finally, it makes translation and dubbing easy, as explained in Sect. 13.2.7.

The objectives of a tutorial should be stated at the beginning. The version of the software and the underlying operating system should be stated. Explicitly stating the prerequisite tutorials, if any, although useful, will hard code the requirement and will call for changes if other tutorials are modified, merged, split or deleted. Because of this reason, prerequisite tutorials are mentioned in our web page only.

All sentences should be easy and short to be accessible to the novice and to those who are not fluent in English. We recommend that most sentences be less than 60 characters long. We insist that no sentence exceeds 80 characters. We refer to this as the 60/80 rule. By restricting the length of sentences, we enforce simple and short sentence constructs. Shorter sentences reduce the temporal contiguity effect, thereby improving the learning process (Mayer and Moreno 2003). Another benefit of this restriction accrues during translation and dubbing, as explained in Sect. 13.2.7.

We have a requirement that unless an activity is described earlier in the tutorial or in one of the prerequisite tutorials, it should be stated explicitly. For example, before a button is pressed, we recommend a script of the following type: *Click the button in the bottom right, circular in shape and blue in colour*. In addition to explaining well, this statement allows more time for a beginner to locate the focus area easily. To ensure that this does not become boring, we require that such a detailed instruction appears only once for any topic.

We also encourage the explanation of every activity in detail. For example, we have a rule that the following type of script be avoided: *Go there, click this, etc.* Instead, it should be explained as in the following statement: *Go to top left-hand corner, click file and then menu*.

Checking the mouse or hitting the *enter* key has to be stated explicitly, as novice users often get stuck on these. Moreover, these do not take much time to state: *Click the mouse* or *hit enter*.

Menu items should be stated exactly as they appear in the screen. For example, if the software has `File browse` as a menu item, the script should not address it as `File view`. Although these two have the same meaning, it will slow down the user.

Active voice should be used in preference to passive voice. Direct order of actions should be used. For example, one should not say the following reverse order of actions: *Let us select file from the media option in the main menu*. Direct order also makes the sentences shorter. In addition to giving examples to illustrate an idea, we also encourage the inclusion of common mistakes and a way to address them. Finally, we encourage the scriptwriter to subject every sentence to *Telephone Test*.

We insist that all STs are activity based. About 75–80 % of the tutorial should be demonstration based that a learner can practise simultaneously. The reason is that if there is too much theory, a self-learner at a remote centre may not get any help if they have a doubt. Because of our FOSS restriction, every user can get the benefit of active learning through the side by side method; see Fig. 13.1.

We recommend that theory, if any, follow the activities mentioned above. For example, after explaining how to write a letter through a series of steps, one can

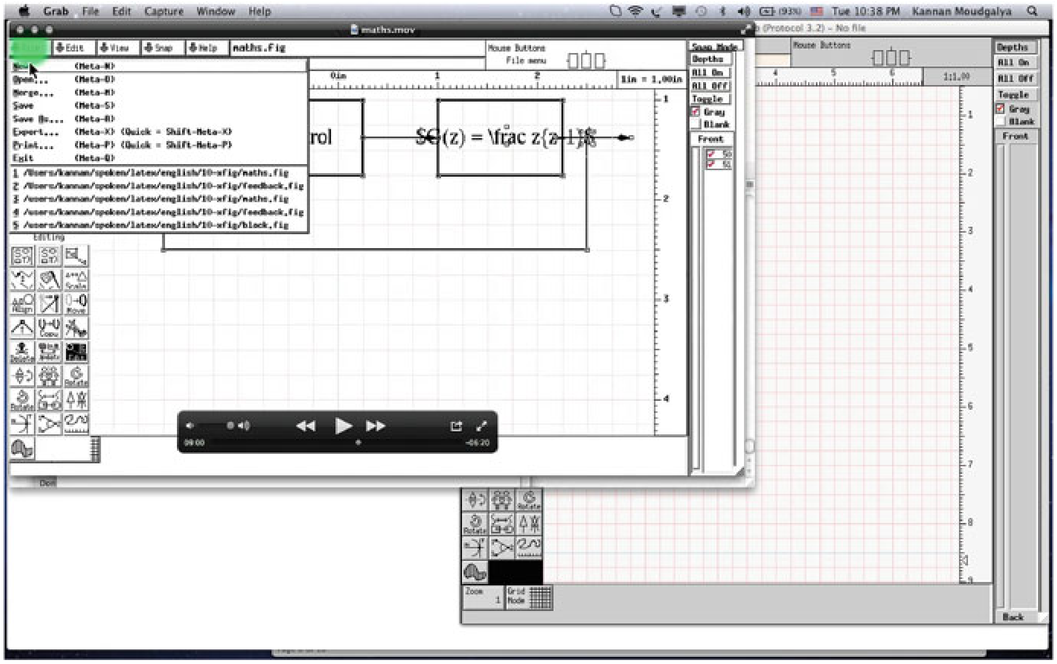
Grab File Edit Capture Window Help
Tue 10:38 PM Kannan Moudgalya
maths.mov
Edit
View
Snap
Help
maths.fig
Mouse Buttons
File menu
New (Meta-N)
Open... (Meta-O)
Merge... (Meta-M)
Save (Meta-S)
Save As... (Meta-A)
Export... (Meta-X) (Quick = Shift-Meta-X)
Print... (Meta-P) (Quick = Shift-Meta-P)
Exit (Meta-Q)
1 /Users/kannan/spoken/latex/english/10-xfig/maths.fig
2 /Users/kannan/spoken/latex/english/10-xfig/feedback.fig
3 /users/kannan/spoken/latex/english/10-xfig/maths.fig
4 /users/kannan/spoken/latex/english/10-xfig/feedback.fig
5 /users/kannan/spoken/latex/english/10-xfig/block.fig
Editing
rol
$G(z) = \frac z{z-1}$
1in = 1.00in
Snap Mode
Depths
All On
All Off
Toggle
Gray
Blank
Front
Back
Zoom
Grid Mode

Fig. 13.1 ST on the FOSS system Xfig is on the *left-hand side*, with the Xfig software itself on the *right-hand side*, both in a reduced and in an unmaximised size. We recommend the learner to listen to a command, pause the video and try it out on the software. As the STs are available for free and we work with only FOSS, this approach is possible for everyone

state the benefits of such an activity: *This is called a letter. A letter is useful to apply for leave, job application, etc.* This goes against the conventional method of presenting the theory first and then demonstrating it through examples or activities.

The 75–80 % rule mentioned above may be difficult to enforce in the first tutorial, however, because one would normally explain the features of a software – there is nothing to practise there. We take care of this through an *Overview Tutorial* that explains subsequent tutorials; see Sect. 13.2.1. If the learner wishes, they can see and practise the latter while going through the former.

We now comment on how the length of an ST is estimated. The script writer reads out aloud the script, without carrying out the indicated activities, and notes down the time required to do this. This time duration is multiplied by a factor to estimate the length of the ST. A typical estimate of this factor is 1.5.

13.2.3 Processes to Ensure Suitability for Self-Learning

We rely on the following steps to make ST suitable for self-learning:

1. Our guidelines to create ST, which are similar to the ones described in Moseley (2013), are given in the following format (ST Team): (1) what the guideline is, (2) why this guideline has been suggested and (3) what will happen if that guideline is not followed.

To ensure that the readers read and understood our rules, we conduct a test based on these guidelines. Only those who score above 75 % in this test are allowed to create STs for us. Those who fail in this test have to wait for 2 weeks before taking the test again. This waiting period forces one to learn the guidelines well before attempting the test and it is a standard practice in learner permit tests.

2. We conduct a novice check of every script before recording it. The details of this step are given in Sect. 13.2.4.
3. The STs go through several rounds of admin review, user review and expert review and corresponding revisions.
4. We conduct pilot workshops to get feedback from a large number of users. Recreation of the STs are required if there are major flaws.
5. Finally, we come up with an instruction sheet for every topic. The details of why an instruction sheet is useful and how it is arrived at are given in Sect. 13.3.3.

13.2.4 Novice Check

To ensure the suitability for self-learning, we enforce a *novice check* on scripts before creating STs. A novice check allows a beginner to verify whether they can reproduce every step of the tutorial. If a script passes a novice check, it is permitted to be recorded, else, it is sent for improvement.

At the beginning of our work, we were worried that the STs would end up boring because of detailed explanation of each and every thing. Fortunately, when asked to rate how interesting the STs were, the following numbers of population reported in Table 13.1 responded as very bad, bad, fair, good and very good, respectively: 182, 440, 4,241, 12,073 and 8,140.

Most of the people who undergo our workshops are beginners, as can be seen in Sect. 13.4. Novice check ensures that STs are accessible to our clientele, resulting in many of them becoming experts, one of the objectives of the ST project.

One who does novice check learns to use the underlying software. As we give an honorarium to do a novice check, we must be the only project that pays beginners to learn. More importantly, we give respectability to a novice.

13.2.5 Recording

Using the visual cue and the narration given in the script, recording has to be done, by speaking aloud at a correct speed. To a question whether our tutorials are of correct speed, 504 of the population described in Table 13.1 think that it is slow, 23,185 think that it is of appropriate speed and 1,387 think that it is fast.

In order that everyone can participate in the creation process, we restrict the recording software to FOSS. For the same reason, we do not insist on high-quality, soundproof, recording studios for recording. We have guidelines that everyone can follow to obtain good recording in normal environments: *Close the doors and the*

Table 13.1 Sample distribution: number of girls and boys who gave their feedback state-wise (*left*) and FOSS-wise (*right*)

State	Girls	Boys
Andhra Pradesh	765	1,035
Arunachal Pradesh	13	26
Assam	2	7
Bihar	17	7
Chhattisgarh	208	165
Delhi	85	152
Goa	31	37
Gujarat	2,421	2,991
Haryana	356	533
Himachal Pradesh	60	120
Jammu and Kashmir	17	35
Jharkhand	11	41
Karnataka	611	760
Kerala	287	185
Madhya Pradesh	948	955
Maharashtra	1,827	2,090
Manipur	0	4
Meghalaya	52	0
Mizoram	3	3
Orissa	55	112
Pondicherry	103	78
Punjab	327	272
Rajasthan	524	688
Sikkim	1	6
Tamil Nadu	2,173	1,670
Tripura	34	73
Uttar Pradesh	756	773
Uttarakhand	40	33
West Bengal	200	298
Total	11,927	13,149
FOSS	Girls	Boys
Advanced C++	25	27
C and C++	3,102	3,744
Java	1,050	994
Java Business Application	6	4
KTurtle	14	9
LaTeX	525	460
LibreOffice Suite Base	3	1
LibreOffice Suite Writer	6	35
Linux	2,978	2,636
Linux Ubuntu	690	570
NetBeans	10	7

(continued)

Table 13.1 (continued)

FOSS	Girls	Boys
OpenFOAM	6	17
PHP and MySQL	1, 978	2, 624
Python	441	448
Scilab	1, 093	1, 572
Spoken Tutorial Technology	0	1
Total	11, 927	13, 149

windows; remove the battery charger; turn off the fans; turn off the mobile phone – it is not enough if you keep it in silent mode, etc. Such an egalitarian method is recommended for collaborative content creation in general (Schleicher et al. 2013).

We recommend that a tutorial be recorded in one sitting to ensure uniform tone is maintained. In case a mistake happens in the middle of a sentence, we suggest that the whole sentence be re-recorded to make editing easy.

The recording dimensions of our tutorial are 800 × 600. This helps the ST readable in the side by side method shown in Fig. 13.1 or in low-cost tablets, such as Aakash (Moudgalya et al. 2013). If instead, one records an entire screen and shrinks it a lot to watch it, letters in the ST may become too small to read. For the same reason, we recommend a minimum font size of 28 pt, whenever possible in all typing, on editor and console; the cursor or mouse pointer also has to be enlarged (ST Team).

We suggest that the recording speed be about 4–5 frames per second to keep the recorded file size small. Keeping the frame rate small does not affect most of our tutorials, as we generally do not have fast moving content. The average size of an ST is less than 1 MB per minute.

13.2.6 Rescripting and Timing

After recording, editing, revision, etc., we freeze the recording. At this point, the script is modified to match the spoken sentences, as it is not always possible to speak exactly what is said in the script. We also permit minor changes to the script for dubbing purposes, provided the modified sentence is no longer than the original sentence. This can take care of minor mistakes and unclear explanation, if any. At this point, we freeze the script.

Once the script is frozen, we note the time when each sentence starts. A script in which the visual cue has been replaced by timing information is called the *timed script*, which is useful in many ways:

1. Including the time when a sentence begins can act as a *tag* for video players that can start a video from a selected point. This will allow the learners to go through the script and quickly come to the location they want to see.

2. One can scroll the script while streaming the ST. This may be useful to the hearing impaired or to ensure better understanding (Mitterer and McQueen 2009).
3. This facility is useful for dubbing purposes, as explained in Sect. 13.2.7. We need to get timed script of the original only once, irrespective of the number of languages in which we do the dubbing. This is extremely useful for countries, such as India, that have many languages.

13.2.7 Translation and Dubbing

We dub the spoken part of an ST into all Indian languages, while the video remains in English. We do not allow the original video to be changed. This requires that the dubbed sentences fit within the time taken by English, known as the *available time*. The 60/80 character rule in Sect. 13.2.2 that helps enforce simple English sentences in the original video helps fit the translation into the available time, irrespective of the target language. If on the other hand we permit complex sentences, it is not clear whether the time constraint requirement will be fulfilled. Even after the 60/80 rule, if a translation does not fit into the available time, we allow dropping of words that are not vital. As the spoken instructions are supported by a detailed video (see Sect. 13.2.2), approximations in a few sentences do not create much difficulty. Finally, the sentences spoken without any screen activity can be combined for timing purposes.

The main objective of the dubbing exercise is to teach IT skills to children who are weak in English. This allows us to make concessions while translating: dropping of unimportant words, approximating translation, mixing of words from other languages, etc. This allows us to choose a simple to understand translation amongst all the possibilities. If instead, we had chosen the objective of our work as the promotion of local languages, we could not have taken these liberties. We also recommend that the technical words be not translated. This is to increase the acceptance of learners of ST in the IT industry. When local language equivalents are popular, we use both English and local language versions. We accommodate such explanations in the *instruction sheet*, to be explained in Sect. 13.3.3.

If we allow the original video to be changed at the time of translation, the work can increase enormously: The pedagogy has to be checked again, pilot workshops have to be repeated, etc. Dubbing only the spoken sentences allows us to create an equivalent audio and to stitch with the original video. It takes only about 5 % effort to create a dubbed tutorial using the methods mentioned here, compared to the creation of a original tutorial.

Dubbing involves reading out the translation, with a small gap between sentences and recording the audio, using a software, such as *audacity* (Audacity). We recommend that recording be done in one sitting to ensure consistency of the tone between different parts of the recording. To complete the dubbing, we just have to cut and slide each sentence to the correct time in the timed script, explained in

Sect. 13.2.6. In some audio recording software, we have to begin this audio editing from the last sentence and go backwards.

Our approach of one video and different audio recordings helps store STs in an efficient manner. This should also help if any sentence in the original video has to be changed, either to correct a mistake or when the version of the target software undergoes a major change.

Getting training in English video helps the learners get jobs in the IT industry, including in transnational companies. Given that most Indians have some exposure of English, this does not create difficulties even if one is weak in it. This also allows students to use the commonly available computer systems, software and keyboards. Carrying out everything in a local language calls for alternate, often more expensive, solutions.

Translation work has to be done by mature people who understand the objective of translation, namely, to help teach children who are not comfortable in English, often from poor families and from rural areas. Professional translators often insist on exact translation and refuse to use foreign words, even if the translated sentences are difficult to understand or longer than the original sentences.

While translation has to be done by mature people, dubbing can easily be done by younger people, often students. The latter group is more comfortable about the underlying technologies, such as audio recording, changing the locations of sentences and stitching this audio with the original video.

A person who dubs a tutorial is required to read out the translated script first and familiarise themselves with unusual or hard to pronounce words. They should also ensure that all translated sentences can be narrated in the available time. If any translated sentence does not fit into the available time, they should not speak that sentence faster. Instead, they should contact the translator and ask them to rephrase the sentences. They should do the same if it is difficult to understand any translated sentence. Unfortunately, the dubbing people, who are often younger than the translators, do not carry out this important exercise, resulting in the following difficulties: (i) The audio-video sync gets affected. (ii) Audio-video sync is maintained, but some sentence narrations are faster than the rest of the script. We were forced to withdraw many dubbed tutorials in the past because of this difficulty. Checking the audio-video sync can be automated, but it is not clear whether a change in speed can be detected automatically.

The method outlined in this section has also helped us dub popular ST like instructional material created by others into our languages, including in *Indian English* (Alex; Rolf. Steinort).

13.3 Using Spoken Tutorials

Recent studies have shown that Screencasts are beneficial when students use them and also perceive them to be useful (Green et al. 2012). This motivates us to conduct workshops, even though the STs are already suitable for self-learning. We detail our approach in this section.

13.3.1 Side by Side Method to Use Spoken Tutorials

An ST can be thought of as a demonstration of a series of commands. The learner should open the ST in a reduced size and open the corresponding software side by side and practise; see Fig. 13.1.

Because we restrict to only FOSS systems and because the STs are available for free download, this is always possible.

The method suggested above is supported by educational psychology research. In a classic paper, Chandler and Sweller (1991) assert that one should keep the cognitive load low for effective learning. The mental load will definitely be higher if a learner does not follow the method suggested in Fig. 13.1, but watches an ST in a full-screen mode and then switches to the software to try out the commands. Hegarty and Just (1989) have shown that the learning objects should be kept side by side to reduce the eye movement, thereby improving the learning effectiveness. In another classic paper, Mayer and Moreno (2003) argue that as the size of working memory is small, one should put less load on it. If a beginner does not follow the method suggested in Fig. 13.1 and hence has to change the screen after every instruction, the load on the working memory is indeed high.

Two senior professors in an engineering college who underwent an ST-based workshop on LaTeX have the following to say about this method:

> The way Prof. Kannan teaches is excellent. Hands-on practice is important. In the side by side method, we listen to the video and practise. This method is useful and efficient for beginners...
> *Prof. Kavita Thakur, Dept. of Electrical Engineering, Govt. College of Engineering, Amravati*

> Being a person from Civil Engineering, I always had a phobia that all learning using technology methods are for IT experts. But this course broke down this notion. You brought all the required technology on my PC and laptop and these Spoken Tutorials are as if I have my teacher with me round the clock to help me learn...
> *Prof. Kshitiza Kadam, Dept. of Civil Engineering, Govt. College of Engineering, Amravati*

As working with maximised screens has become a norm nowadays, especially in MS Windows-based systems, most people are not familiar with reduced screens that are neither maximised nor minimised. We refer to such reduced screens as *unmaximised* screens. Use of maximised screens was possibly justified in early days of PCs when the screen size was small. Given that larger-sized screens are available, it is easy to follow the suggested method. The guidelines on font, cursor size, etc., explained in Sect. 13.2.5 and in ST Team, ensure that one can read the contents of STs, even after shrinking.

As an ST is created for self-learning, every command can be executed by the learner. The procedure to practise is simple: Listen to a command; pause the video; try it out on the software; if it works, proceed with the next command; if not, rewind the video and listen to the command once again. Repeat this until the whole ST is completed. As every command has to be repeated, our method promotes active learning.

13.3.2 Motivation for SELF Workshops

As STs are created for self-learning, it is possible to conduct workshops without domain experts. In this section, we describe SELF Workshops, the word SELF being an acronym of Spoken Tutorial based Education and Learning through free FOSS study. Although STs are available for download, there are many benefits in organised workshops:

1. If a college agrees to the idea of SELF workshops, it may organise them during class hours and allow their students to participate in them. Independent access to Internet may not be available to the students during college hours, otherwise.
2. Training is possible also to students who do not have their own laptops – the college infrastructure can be used for this purpose.
3. The conductor of the workshop at the college can interact with the ST team and ensure that all the systems in their computer lab are hardware and software ready before the start of a workshop.
4. If a college administration is convinced of the efficacy of SELF workshops, it can make this training mandatory on some topics, for example, C and Java for their curriculum or for placement.
5. Colleges provide a convenient environment to conduct online tests. Moreover, a college teacher can proctor the tests. We conduct online tests only under the proctoring mode.
6. When a college uses STs through SELF workshops or otherwise, there is an added authenticity and increased acceptance.

Internet is not required at the time of SELF workshops – it is required only at the time of online tests. Before conducting a SELF workshop, the conductor of the workshop gets all the required ST either by downloading from our website or through CDs from our team and copies them on each computer system for offline use. A similar method is applicable for the target software as well. Such a coordination helps facilitate the conduct of online tests as well. It is easy to scale up this activity: From the scrolling list of workshops on Spoken Tutorial Project, we expect to conduct about 10,000 SELF workshops in this year.

13.3.3 Instruction Sheet

A set of Spoken Tutorials has to be studied if one wants to learn a FOSS. For example, one needs 10 STs to learn LaTeX, 9 to learn Linux, 17 to learn C and so on. We use *instruction sheets* to convey this information.

An instruction sheet contains the following information: (1) It lists the tutorials to be studied for a particular FOSS and the sequence of study. (2) It gives additional instructions or tasks to be carried out. For example, it may state the following: *At 5:45 min, reproduce the calculations shown in the tutorial.* (3) It may give additional explanation of the following type, which may have been missed out in the video: *At 3:23 min, the author completes the explanation of how a source file is created. Its use will be explained now*. (4) It may clarify some ambiguities, for example, it may say the following: *Note that* `pdflatex` *is one command; there is no space between* `pdf` *and* `latex`. (5) Minor mistakes in the tutorial, which do not warrant a correction in the video can be addressed. Here is an example of it: *At 7:42 min, read as colon operator, although the ST says semicolon operator*.

We arrive at instruction sheets through pilot workshops (Moudgalya 2011b). One begins a pilot workshop with an initial draft of the instruction sheet, based on the initial vision of the ST creation team. The instruction sheet is updated based on the difficulties and the doubts the students have. The help feature for the `pdflatex` command explained above is based on the difficulty faced by the students, for example. The instruction sheet is improved until all possible difficulties associated with the target STs are sorted out. If major flaws are found, however, we recreate the STs.

The set of guidelines for the first video in an instruction sheet is usually elaborate as many doubts, including the obvious ones, have to be answered. The guidelines for subsequent tutorials need not be detailed and one may reduce them gradually or steeply, depending on the pilot workshops. A sample LaTeX instruction sheet is available at ST Team.

13.3.4 Duration of SELF Workshops

The SELF workshops are of 2-h duration. During this workshop, a student learns how to use an ST. They also practise three or four STs, each of which is of 10-min duration. If the students are interested, they can copy all the STs on to a storage device, such as a pen drive, and study at home. They may appear in an online test, which may be organised about 2 weeks after a SELF workshop is conducted.

One of the reasons for choosing a duration of 2 h for a SELF workshop is to reduce the organisational cost and to make the computer facility available to a larger number of students. For example, it may be possible to conduct a SELF workshop every evening or three workshops on holidays. As these workshops are conducted free of cost, it is important to cut down losses – reduce the time spent on uninterested students, who may have participated in the SELF workshops for various reasons. The organisers also need not give anything to eat or drink in 2-h workshops.

13.3.5 Conductor of SELF Workshops

SELF workshops are conducted by a volunteer, a student or a staff member of a college, referred to as a *conductor*. As the STs are created for self-learning and the instruction sheets further explain what to do, it is not required for a conductor to be a domain expert at all. The role of the conductor of a SELF workshop is to enforce discipline and to help convey the method of using STs.

We will illustrate the role of a conductor with the help of two examples. Many students who appear in a Linux workshop do not know the difference between a terminal and an editor. Some of them type a command on a terminal and wait for an answer. The conductor of the workshop should point out the mistake and ask the student to start the tutorial from scratch. In a Linux workshop, some commands are to be executed in the background with the help of ampersand (&) operator. The participants who forget to use & will complain that the subsequent commands do not work. The conductor should point out the mistake and ask the students to start the tutorial from scratch, once again.

The reason for asking the students to start from scratch in case of mistakes is that retrieving from a mistake could require domain expertise. The conductor, who is not necessarily a domain expert, may not be in a position to address this issue. Moreover, if the conductor starts addressing such issues, they will not be able to handle large numbers of students and the 2-h duration could become insufficient. Finally, the time a student has to spend extra in case of a repeat listening is about 5 min, which is not much. At the end of a SELF workshop, a student will realise that if they follow exactly what is shown in an ST, everything will work correctly.

Because of the generality of the suggested procedure, a conductor is in a position to conduct SELF workshops on other topics also, after successfully completing one. This is an important factor that has helped us scale up the number of SELF workshops.

13.3.6 Learning in a SELF Workshop

The SELF workshops are somewhat dictatorial in nature, with the learning outcome to match the contents of STs. Given that there is no support of domain experts in the SELF workshops, the students are expected to do only the things suggested in the STs. The students can, of course, try the things not covered in a tutorial. If they get stuck, the conductor may advise the students to close the current ST and to move on to the next one.

If we do not encourage students to try out things on their own, do they learn anything in a SELF workshop? It turns out that they learn quite a bit. For example, most students learn the following in a $\LaTeX$ SELF Workshop: (i) What is meant by compiling? (ii) How to write letters in $\LaTeX$? (iii) How to write reports? (iv) How to write mathematical equations? (v) Introduction to presentation using Beamer (Moudgalya 2011b). To the best of our knowledge, though, no other method can a student learn so much in 2 h. In other words, for most students, what we propose for a workshop is new anyway and hence it makes no sense to try out something else that we do not cover and for which no help may be available. We have recently implemented an online forum to answer the questions a student may have in STs (Spoken Tutorial Project).

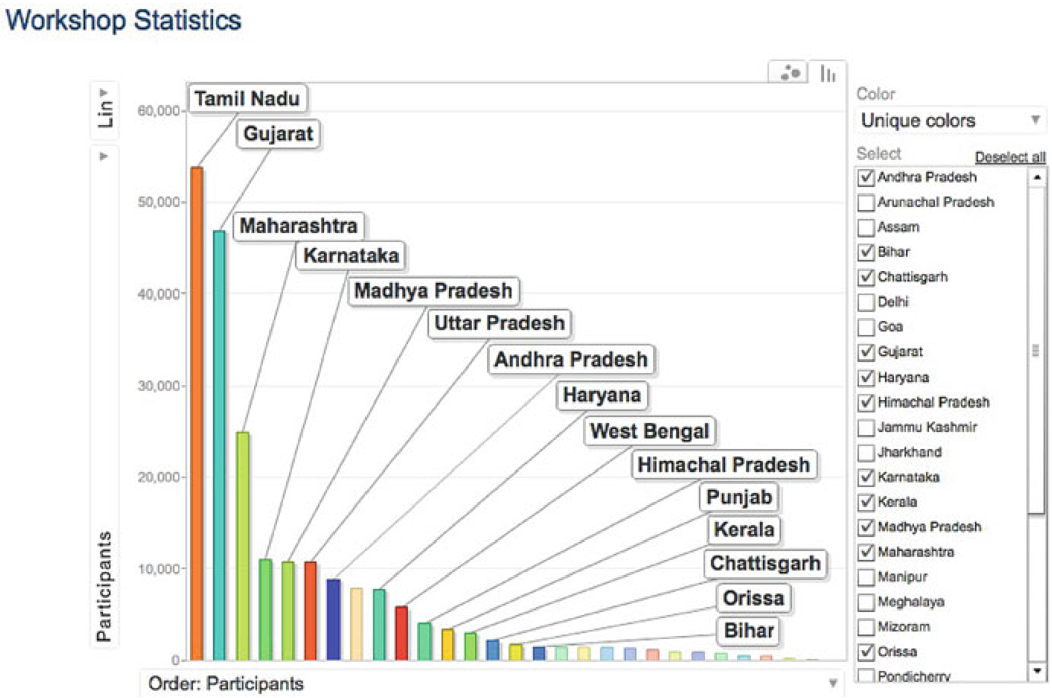

Fig. 13.2 Number of students who have undergone SELF workshops in different parts of India, as of 11 Feb. 2014

13.3.7 Other Support Services

In this section, we briefly discuss the web and online testing support. Every ST can be downloaded from our website free of cost and without any registration. In addition, we provide the following to support the users of ST: Outline, Script and Timed Script. In addition, all the files required to practise with an ST are provided. For example, along with the LaTeX Tables and Figures ST, we provide the following files: `tab-fig.tex`, `iitblogo.pdf`, `iitb.pdf` and `cclicesses.sty`. Similarly, for the C and C++ Function Call ST, we provide the files `callbyval.c` and `callbyref.c`.

About 215,000 college students have undergone SELF workshops until now. Figure 13.2 gives a statistics of the number of students who have undergone SELF workshops in different parts of India. We conduct online tests for students who have gone through ST and give certificates for those who score passing marks. More than 50,000 students have undergone online tests with about 75 % pass rate. We are in the process of implementing programming-type questions with a requirement that a student should get at least one working code to pass a test.

The pedagogical and the effectiveness aspects of our approach have been summarised in Eranki and Moudgalya (2012a,b,c, 2013a,b).

13.4 Testimonials and User Feedback

In this section, we present the user feedback received from about 25,000 of the participants of SELF workshops. We now collect feedback from only those participants who have access to the Internet. It is interesting to note that only about a fourth of all the SELF workshop participants undergo our online tests. Although some students undergo online tests through Internet browsing centres that can be hired for this purpose, it is an exception.

In Table 13.1, we show the number of boys and girls who gave their feedback on the basis of the state they are from and the FOSS topic of their SELF workshop. One can see that the respondents are from a wide geographical area and their inputs are on different FOSS topics.

The participants of SELF workshops, as distributed in Table 13.1, gave the following feedback. When asked about the quality of instructional material used in the SELF workshops, the response was as in Fig. 13.3. One can see that more than 80 % of the respondents thought that the quality was good or very good. When asked how good the college computer infrastructure was, we received the response as in Fig. 13.4. Only 60 % of the respondents felt that it was good or very good, with about 30 % of them feeling that it was just fair. This is of course expected as all sorts of colleges, including those that do not have a good infrastructure, have taken part in the SELF workshops. When asked how good the conductor of the workshop in their college was, we received the response as in Fig. 13.5. More than 80 % of the respondents feel that their workshop organiser was good or very good. As the same person can conduct workshops on many topics, the conductors become good quickly. When asked how they would rate the mentor from the Spoken Tutorial Team at IIT Bombay who came on the Skype and talked to them, the feedback was as in Fig. 13.6. Once again, 80 % feel that they are good or very good. When asked about the overall quality of the workshop, we received the feedback as in Fig. 13.7. When asked about their perception of the applicability of the FOSS system that they learnt, they gave the reply as in Fig. 13.8. Although about 75 % of the students think

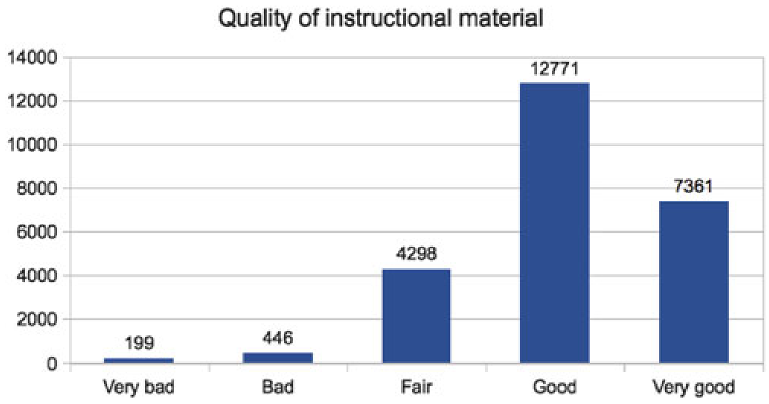

Fig. 13.3 Feedback on the quality of instructional material

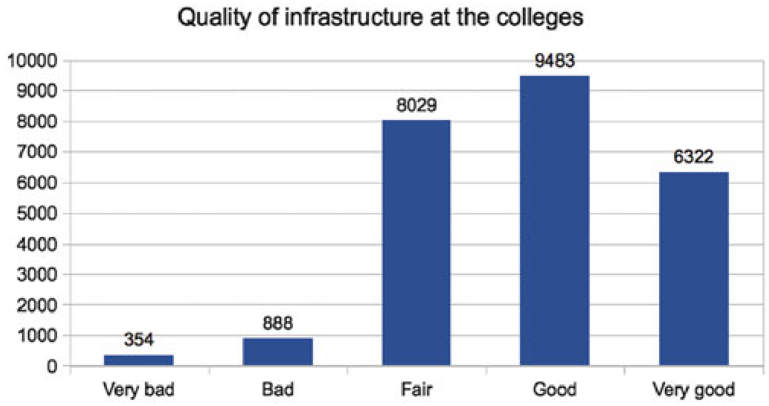

Fig. 13.4 Feedback on the quality of the computer infrastructure in the colleges

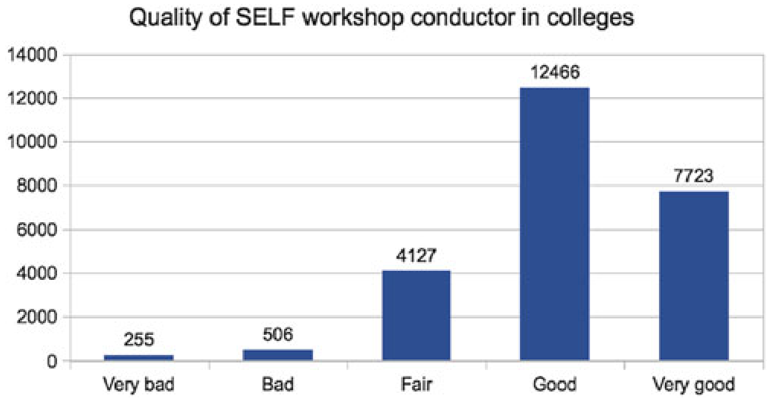

Fig. 13.5 Feedback on the quality of the conductors of the workshop in the colleges

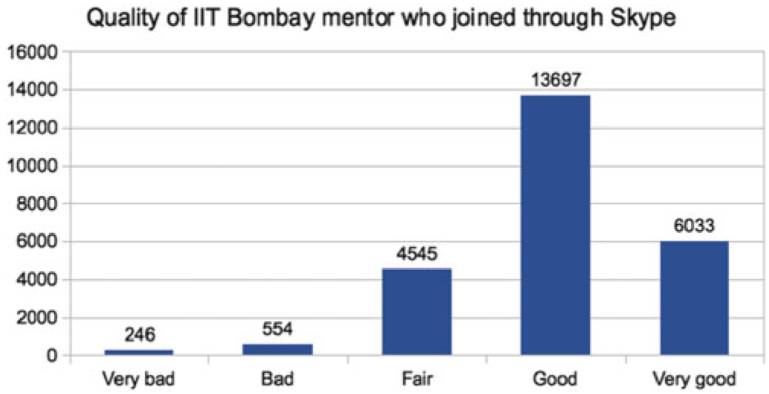

Fig. 13.6 Feedback on the quality of the mentor from the Spoken Tutorial Team at IIT Bombay who joined through Skype

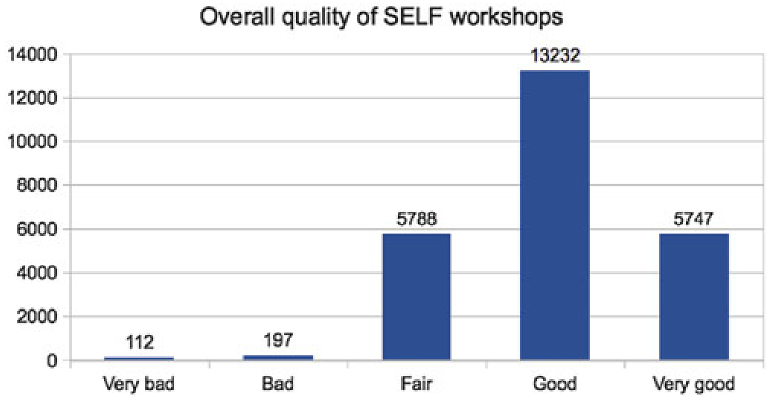

Fig. 13.7 Feedback on the overall quality of the workshop

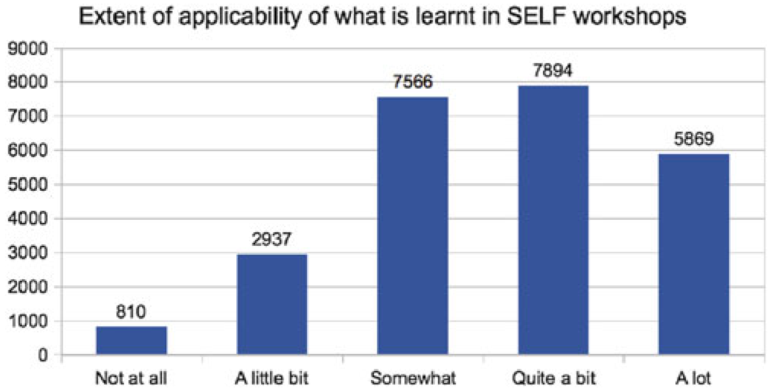

Fig. 13.8 Feedback on the applicability of what is learnt in SELF workshops

that the overall quality of the SELF workshop is good or very good, just a little over 50 % of the students only feel that what they learn is applicable quite a bit or a lot. This shows that curiosity also plays a role in participating in these workshops. Nevertheless, about 85 % of the people feel that they are likely to or quite likely to or definitely will recommend this methodology to others, as can be seen from Fig. 13.9.

It is no wonder that universities and government departments have started adopting the ST methodology in their curriculum. For example, Himachal Pradesh (HP) University has included the ST material under the Choice Based Credit System for its 77 affiliated colleges (HP University 2013). The B.Sc (IT) students of HP University have to do one three-credit course from the following four, with the instructional material being in the form of ST: (i) Basic IT-based skills (Linux, LibreOffice, Firefox and Tux typing), (ii) Blender, (iii) GIMP and (iv) Java. The B.Sc (CS) students of HP University have to do one three-credit course from one of the following four, using the ST instructional material: (a) Python, (b) Scilab, (c) C and C++ and (d) PHP and MySQL. The Tamil Nadu Department of Technical Education is in the process of introducing the ST methodology to all the about 450 polytechnic colleges in the state (Tamil Nadu Dept. of Technical Education 2013).

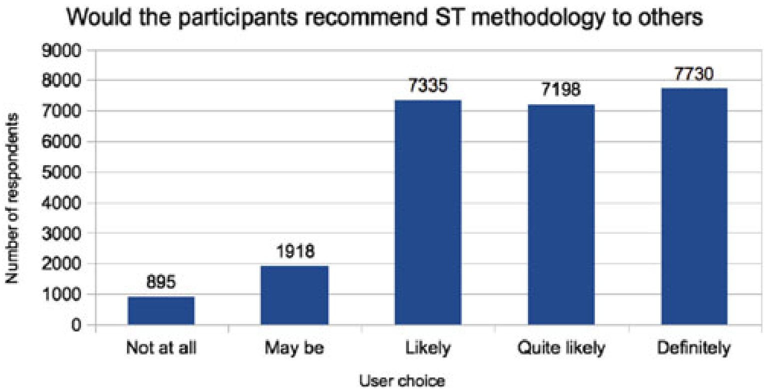

Fig. 13.9 Feedback on whether they will recommend the ST and the SELF workshop methodology to others

13.5 Conclusions and Future Work

The ST project, which began as a documentation effort for FOSS, has turned out to be a full-blown training programme. We have received a large number of testimonials from students and teachers that claim that the ST methodology helps (i) self-learn IT topics, (ii) improve job opportunities, (iii) learn FOSS tools better than their commercial counterpart and (iv) reach even those who are weak in English without affecting the employment potential in the IT sector.

We are now working with officials who coordinate universities, engineering colleges, polytechnic colleges, schools and citizen centres to spread the IT literacy. To this end, our team takes care of the following activities: (a) mapping the academic content, (b) creating study material, (c) providing access to online tests and (d) training the instructors.

Using ST, more than 100,000 students are trained in a year through the SELF workshop mechanism in one FOSS or another. The number of page views of our portal (Spoken Tutorial Project) has been doubling every year. We expect about three million page views in the current year, with the average time spent being about 8–9 min (Awstats). We have received positive feedback from more than 25,000 users about the efficacy of the ST methodology. We are receiving a large number of testimonials from the satisfied students and teachers (Spoken Tutorial Team) about the different benefits that they have received through the ST methodology.

We are in the process of building forum-based support to answer student questions (Spoken Tutorial Project). We have enabled discussions on the basis of the timeline of a tutorial. Although this will make it difficult to replace a tutorial, we believe that this is the most friendly way for students to self-learn and get individualised help through ST. Hopefully, this will also help us address the substantial changes when a new version of the software is released.

We are now working on methods to ensure that the users of STs learn properly so that most of them can pass our online tests. Correlating the learning and the actual benefit is also on our radar. Finally, *bridging the digital divide* is one of the ultimate goals of this project, although not much has been achieved so far.

The National Mission on Education through ICT (Ministry of Human Resource Development) which funds the ST effort also funds the world's lowest-cost access device, Aakash (Moudgalya et al. 2013). Given that it can be used as a full-blown computer system (Patil and Patnaik 2013) and that a lot of applications are already ported (Aakash Team), Aakash can be used to spread IT literacy. Without such an inexpensive but a powerful access device, making IT literacy available to every Indian would be next to impossible.

Acknowledgements The author wants to thank Nancy Varkey and Shyama Iyer for leading the efforts in the creation and the use of ST, respectively. He thanks Vishnu and Sanmugam for the web support and in collating the user feedback. He thanks Kiran Eranki for carrying out many useful research studies on the use of ST. The author wants to thank the entire FOSSEE (http://fossee.in) team for supporting the ST efforts. Finally, he thanks the funding agency, National Mission on Education through ICT (Ministry of Human Resource Development), for a generous financial grant.

References

Aakash Team. A Brochure on Aakash applications. See http://aakashlabs.org/builds/Aakash_brochure_inside_20_11_2013.pdf and http://aakashlabs.org/builds/Aakash_brochure_outside_20_11_2013.pdf. Seen on 11 Feb 2014.

Alex. PHP Academy. See https://phpacademy.org/. Seen on 11 Feb 2014.

Audacity. Audacity is free, open source, cross-platform software for recording and editing sounds. See http://audacity.sourceforge.net/. Seen on 11 Feb 2014.

Awstats. Analytics provided by awstats on spoken-tutorial.org. See http://spoken-tutorial.org/awstats/awstats.pl?config=www.spoken-tutorial.org. Seen on 11 Feb 2014.

Chandler, P., & Sweller, J. (1991). Cognitive load theory and the format of instruction. *Cognition and Instruction, 8*(4), 293–332.

Eranki, K. L. N., & Moudgalya, K. M. (2012a). Evaluation of student perceptions and interests using spoken tutorials in online courses. In *International conference on advanced learning technologies, ICALT*, Rome, 4–6 July 2012 (pp. 308–310). IEEE.

Eranki, K. L. N., & Moudgalya, K. M. (2012b). Evaluation of web based behavioral interventions using spoken tutorials. In *Technology for education, T4E*, Hyderabad, 18–20 July 2012 (pp. 38–45). IEEE.

Eranki, K. L. N., & Moudgalya, K. M. (2012c). A collaborative approach to scaffold programming efficiency using spoken tutorials and its evaluation. In *CollabCom*, Pittsburgh, Oct 2012 (pp. 556–559). IEEE.

Eranki, K. L. N., & Moudgalya, K. M. (2013a). Application of puzzles to unpuzzle the programming difficulty of students through spoken tutorial workshops. In *The 21st international conference on computers in education*, Bali, 18–22 Nov 2013. IEEE.

Eranki, K. L. N., & Moudgalya, K. M. (2013b). An integrated approach to build programming competencies through spoken tutorial workshops. In *International conference on technology for education T4E*, Kharagpur, 18–20 Dec 2013. IEEE.

Farrell, D., Kaka, N., & Sturze, S. (2005). Ensuring India's offshoring future (pp. 75–83). New York: McKinsey & Company.

Green, K. R., T. Pinder-Grover, & Millunchick, J. M. (2012). Impact of screencast technology: Connecting the perception of usefulness and the reality of performance. *Journal of Engineering Education, 101*(4), 717–737.

Geron, T. (2013). Bill Gates, Mark Zuckerberg, Chris Bosh Campaign for more programmers. In: *Forbes*, 26 Feb 2013. http://www.forbes.com/sites/tomiogeron/2013/02/26/bill-gates-celebrities-support-education-for-computer-programming/. Seen on 11 Feb 2014.

Guo, P. (2013). Optimal video length for student engagement. See https://www.edx.org/blog/optimal-video-length-student-engagement. Seen on 11 Feb 2014.

Hegarty, M., & Just, M. A. (1989). Understanding machines from text and diagrams. In H. Mandl & J. R. Levin (Eds.) *Knowledge acquisition from text and pictures* (Advances in psychology, Vol. 58, pp. 171–194). Amsterdam: North-Holland.

HP University. (2013). Circular issued to all affiliated colleges. See http://hpuniv.nic.in/pdf/CBCS/cbcs_workshop13.pdf and http://hpuniv.nic.in/pdf/Spoken_Tut13.pdf. Seen on 11 Feb 2014.

Koufogiannakis, D., & Wiebe, N. (2006). Effective methods for teaching information literacy skills to undergraduate students: A systematic review and meta-analysis. *Evidence Based Library and Information Practice, 1*(3), 3–43. Available at http://eprints.rclis.org/8343/. Seen on 11 Feb 2014.

Mayer, R. E., & Moreno, R. (2003). Nine ways to reduce cognitive load in multimedia learning. *Educational Psychologist, 38*(1):43–52.

Ministry of Human Resource Development. National mission on education through ICT. http://www.sakshat.ac.in. Last seen on 11 Feb 2014.

Mitterer, H., & McQueen, J. M. (2009). Foreign subtitles help but native-language subtitles harm foreign speech perception. *PLoS One, 4*(11):e7785. doi:10.1371/journal.pone.0007785.

Mohorovičić, S. (2012). Creation and use of screencasts in higher education. In *MIPRO 2012: Proceedings of the 35th international convention*, Opatija (pp. 1293–1298). IEEE.

Moseley, S. (2013). *Getting started with screencasting: A tool to supplement classes, answer student questions, and provide guided analysis practice* (Number 6491 in 120th ASEE annual conference and exposition, Atlanta, 23–26 June 2013). American Society for Engineering Education.

Moudgalya, K. M. (2009). Spoken tutorials. In *IEEE technology for education, T4E 2009*, Bangalore (pp. 17–23). IEEE.

Moudgalya, K. M. (2011a). Spoken tutorial: A collaborative and scalable education technology. *CSI Communications, 35*(6):10–12. Available at http://spoken-tutorial.org/CSI.pdf. Seen on 11 Feb 2014.

Moudgalya, K. M. (2011b). LaTeX training through spoken tutorials. *TUGboat* 32(3):251–257.

Moudgalya, K. M. (2013). Spoken tutorials – Towards IT literacy. In *gn Y-point, IIT Bombay Alumni Association New York Chapter Magazine* (pp. 9–12), 26 Oct 2013. Available at http://spoken-tutorial.org/GnYPoint.pdf. Seen on 11 Feb 2014.

Moudgalya, K. M., Phatak, D. B., & Shevgaonkar, R. K. (2008). Engineering education for everyone: A distance education experiment at IIT Bombay. In *Proceedings of the 38th annual frontiers in education conference, FIE'08*, Saratoga Springs, 22–25 Oct 2008 (pp. T3C–21–T3C–26). IEEE.

Moudgalya, K. M., Phatak, D. B., Sinha, N. K., & Varma, P. (2013). Genesis of Aakash 2. *CSI communications* (pp. 21–23). 29 Jan 2013. Available at http://aakashlabs.org/media/pubs/genesis-reprint.pdf. Seen on 11 Feb 2014.

Novaković, D., Milić, N., & Milosavljević, B. (2013). Animated vs. illustrated software tutorials: Screencasts for acquisition and screenshots for recalling. *International Journal of Engineering Education, 29*(4), 1013–1023.

Patil, S., & Patnaik, S. (2013). GNU/Linux on Aakash. In *CSI Communications*, July 2013 (pp. 28–31). Available at http://aakashlabs.org/media/pubs/GNU_Linux_on_Aakash.pdf.

Rolf. Steinort. Meet the GIMP. See http://blog.meetthegimp.org/. Seen on 11 Feb 2014.

Schleicher, R., Westermann, T., & Möller, S. (2013). High – vs. low-quality video lectures: Don't worry, just put them online! In *CSEDU 2013 – Proceedings of the 5th international conference on computer supported education*, Aachen (pp. 204–209).

Spoken Tutorial Project. Official web page. See http://spoken-tutorial.org/. Seen on 11 Feb 2014.

Spoken Tutorial Project. Online forum. See http://forums.spoken-tutorial.org/. Seen on 11 Feb 2014.

Spoken Tutorial Team. Testimonials page. See http://spoken-tutorial.org/testimonials. Seen on 11 Feb 2014.

ST Team. Guidelines to create spoken tutorials. See http://process.spoken-tutorial.org/index.php/FOSS_Stages/Checklist. Seen on 11 Feb 2014.

ST Team. LaTeX instruction sheet. See http://spoken-tutorial.org/sites/default/files/st_videos/LaTeX/LaTeX_Instruction_Sheet_English.pdf. Seen on 11 Feb 2014.

Tamil Nadu Dept. of Technical Education. (2013). Circular issued to all polytechnic colleges in Tamil Nadu. See http://tndte.com/spoken%20tutorial.pdf and http://tndte.com/spoken%20tutorial1.pdf. Seen on 11 Feb 2014.

Dr. Kannan M. Moudgalya is a professor of Chemical Engineering, Systems and Control, and Education Technology at IIT Bombay. He has been with IIT Bombay for 25 years. He was a visiting professor at the University of Alberta. He received chemical and electrical engineering degrees from IIT Madras and Rice University. Prof. Moudgalya has written two textbooks: (1) *Digital Control*, published by John Wiley & Sons, Chichester and (2) *Optimization: Theory and Practice*, jointly with Prof. M. C. Joshi, published by Narosa, New Delhi. He has published a large number of papers in refereed international journals and conferences in the areas of mathematical modeling, control, and simulation. Prof. Moudgalya is now devoting his time on spreading education on a massive scale, without quality dilution. He has been focusing on spoken tutorials, open source software systems, virtual labs, and the low cost tablet, Aakash. He has held the posts of associate dean (R&D), head of Océ Automation, and head of the Centre for Distance Engineering Education Programme, at IIT Bombay. He is a member of the Standing Committee of the National Mission on Education through ICT, MHRD, Government of India.