

Interactive Learning Objects

David Rossiter, Gibson Lam and Leo Tsui

Department of Computer Science and Engineering
Hong Kong University of Science and Technology

11th December 2007

Contents

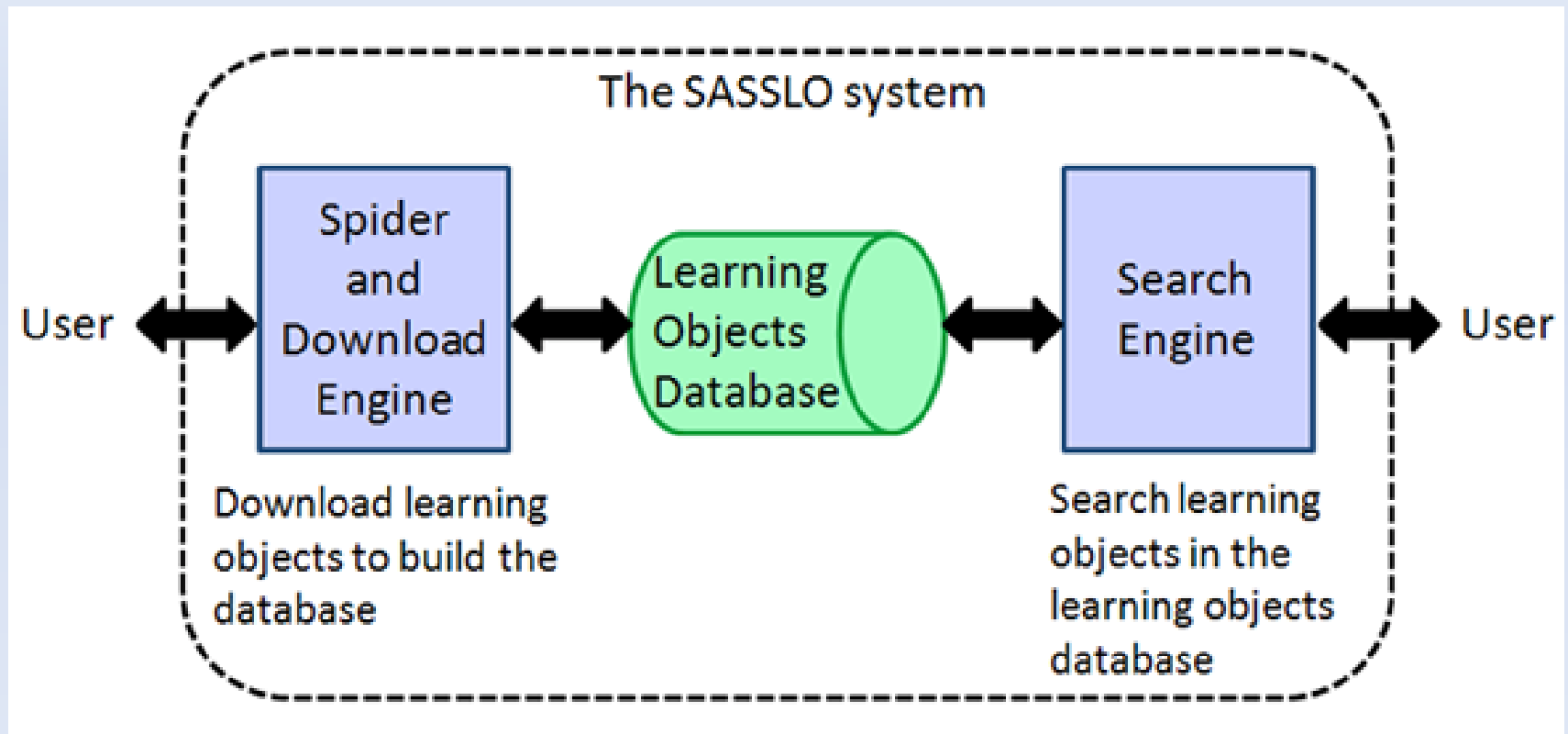
- Introduction
- SASSLO
- Populating SASSLO Database
- Searching SASSLO database
- Developing a Database of Computer Science Learning Objects
- Construction of new learning objects
- Conclusions

Introduction

- A well designed interactive learning object can be extremely useful in teaching
- There are many such learning objects on the Internet
- A system was developed to provide a central repository in which those objects can be easily obtained, and subsequently searched and retrieved

The SASSLO System

- We developed an open source system called SASSLO



Populating SASSLO Database

- Populating a SASSLO database:
 1. User finds website which holds objects of interest
 2. User enters URL and search parameters
 3. System searches multiple levels and grabs everything
 4. Files are presented to the user
 5. User selects useful objects, others are discarded
 6. User enters metadata

Starting a Search

1. User finds website which holds objects of

Breadth First Search Java Applet

How the Applet Works

The BFS tree consists of 9 nodes, each numbered in a BFS traversal route. You may enter the number of the node you wish to reach, and then you may press the 'Start' button to begin the BFS traversal process. The applet will first construct a list of the nodes not visited, then a list of the nodes visited while searching for the correct node. Please read [Applet Instructions](#) for further details.

BREADTH FIRST SEARCH

```
graph TD; 5[5] --- 2[2]; 5[5] --- 6[6]; 2[2] --- 1[1]; 2[2] --- 4[4]; 4[4] --- 3[3]; 4[4] --- 7[7]; 6[6] --- 8[8]; 8[8] --- 9[9];
```

Enter the number you wish to search for:

Search Parameters

2a. User enters URL

1. Enter the link you want to process:

http://www.cs.usask.ca/resources/tutorials/csconcepts/1998_3/BFS/java/index.htm

Search Parameters

2b. User selects search parameters

2. Select which file types/extensions you wish to download:

| | | |
|---|--|---|
| <input checked="" type="checkbox"/> Video Files Selected: (1 / 13) show details | <input type="checkbox"/> Image Files Selected: (0 / 6) show details | <input type="checkbox"/> Audio Files Selected: (0 / 7) show details |
| <input type="checkbox"/> Text Files Selected: (0 / 4) show details | <input type="checkbox"/> Web Files Selected: (0 / 9) show details | <input checked="" type="checkbox"/> Executable Files Selected: (1 / 2) show details |
| <input checked="" type="checkbox"/> Compressed Files Selected: (1 / 4) show details | <input checked="" type="checkbox"/> Developer Files Selected: (1 / 3) show details | <input type="checkbox"/> Miscellaneous Files Selected: (0 / 5) show details |

[Select All Available File Extensions](#) [Load Default File Extensions](#)

Search Parameters

2c. Spider search parameters may be adjusted

3. Add the limit parameters:


| | |
|--|--|
| Maximum mirroring depth: | <input type="text" value="1"/> level(s) |
| Timeout period: | <input type="text" value="300"/> second(s) |
| Timeout period for each file download: | <input type="text" value="5"/> second(s) |

The Results

3. System searches multiple levels and grabs everything

4. Files are presented to the user

5. User selects useful objects, others are discarded

| | |
|---|--|
| Web Page Link: | |
| http://www.cs.ust.hk/image_processing/ | |
| <input checked="" type="checkbox"/> Pick all downloaded objects with selected file extensions in this page? | Object Details: |
| <input checked="" type="checkbox"/> Pick this downloaded object? |  |
| <input type="button" value="Input Descriptions"/> | Filename: negation.png |
| | Original Download Link: Link |
| | Downloaded File Size: 64750 bytes |
| | Try the file in Learning Objects Server: Link |
| | Date added: 5-7-2006 |

The Results

6. User enters metadata

Category:

1. Choose a subject category or add a new subject category.

Or
New Subject Category :
2. Add a sub-category for the subject category.

Or
New Sub-Category :
3.
4. Added Category(s) :

Different Types of Search

1. Simple word search
 - Simplest search
 - Enter a word or multiple words which he/she wishes to search
2. Regular expression search
 - Highly complex queries can be entered
3. Advanced search
 - Search operation can take place on any single/multiple field(s) of the metadata
4. File type search
 - Search for specific types of file that are stored in the database

Developing a Database of Computer Science Learning Objects

- A copy of SASSLO was installed on a server
- More than 500 learning objects related to Computer Science were added to the database
- Each object is associated with one or more COMP course(s)
- This system allows the users to easily search for objects which are related to a specific course

Construction of New Learning Objects

- Complements areas lacking in the database
- Most objects in Java Applet form, work on most systems

Areas of New Learning Objects - Algorithms

Array

The diagram shows an array partitioning process. The array is divided into three partitions: [5, 3, 4], [10, 8], and [7, 9, 1]. A pivot element '6' is shown in a red box above the second partition, and a pivot element '2' is shown in a green box below the first partition.

Algorithm

1. Determine the "unprocessed" portion of the array and set it to be unprocessed partition.
2. **While there are unprocessed partitions with more than one element:**
 - 2.1 Choose a pivot (first element in each partition).
 - 2.2 **Move all elements smaller than the pivot to the left and other elements to the right in the partition.**
 - 2.3 Set the pivot at the appropriate position.
 - 2.4 Divide the recent partition into two new unprocessed partitions at the pivot position, if necessary.
3. Combine all partitions to form the final result.

Description

Continue to find out the elements and swap them if necessary.

Fast

Change Array Program

?

Areas of New Learning Objects – File Systems

Unix Learning Tool

swap ?

The unix shell

```
\:1> ls
f1.txt f2.txt
\:2> mkdir testing
\:3> ls -l
-rwx----- 1 peter      students    Dec 11 09:44 f1.txt
-rwx----- 1 peter      students    Dec 11 09:44 f2.txt
drwx----- 1 peter      students    Dec 11 09:44 testing
\:4>
```

Enter your unix command here

```
\:4> 
```

Your role in this file system is:

- owner
- group member
- others

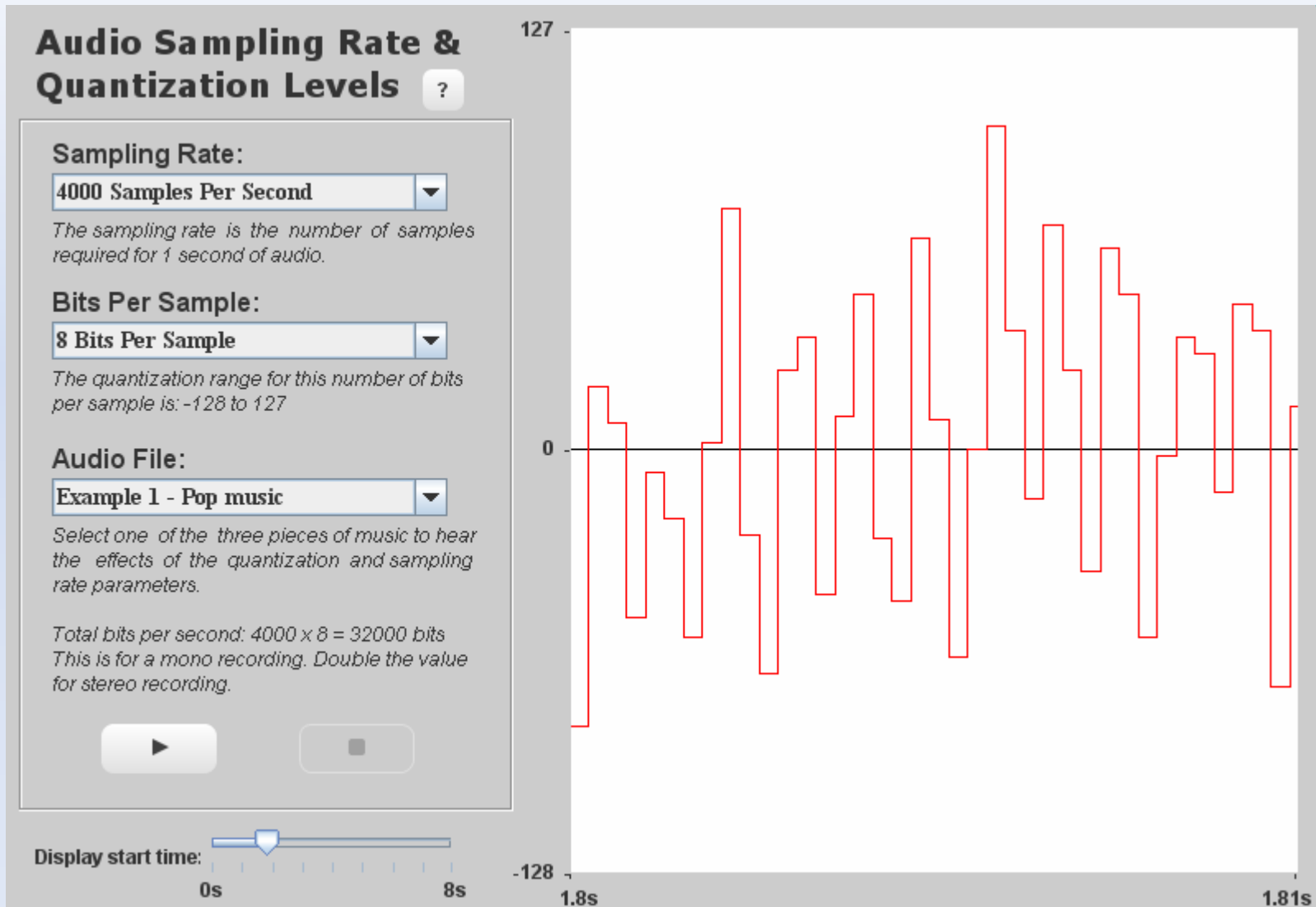
Selected object information

The entire file structure

Show the shortcut buttons

```
graph TD
  root[" / "] --- homes[" homes "]
  root --- f3[" f3.txt "]
  homes --- peter[" peter (current directory) "]
  homes --- f3_file[" f3.txt "]
  peter --- f1[" f1.txt "]
  peter --- f2[" f2.txt "]
  peter --- testing[" testing "]
  testing --- f1_test[" f1.txt "]
  testing --- f2_test[" f2.txt "]
  root --- d2[" d2 "]
  d2 --- f4[" f4.txt "]
  d2 --- f5[" f5.txt "]
  d2 --- d3[" d3 "]
  d3 --- d3_sub[" d3 "]
  root --- d4[" d4 "]
```


Areas of New Learning Objects - Audio



Areas of New Learning Objects - Music

MIDI Sequencer

Please select the instrument:
0 Piano

Volume

Last MIDI message sent -

Type: NOTE OFF

Pitch: 75

Velocity: 0

Instrument: -

Hex. Message: 80 4b 00

Bin. Message: 1000 0000 0100 1011 0000 0000

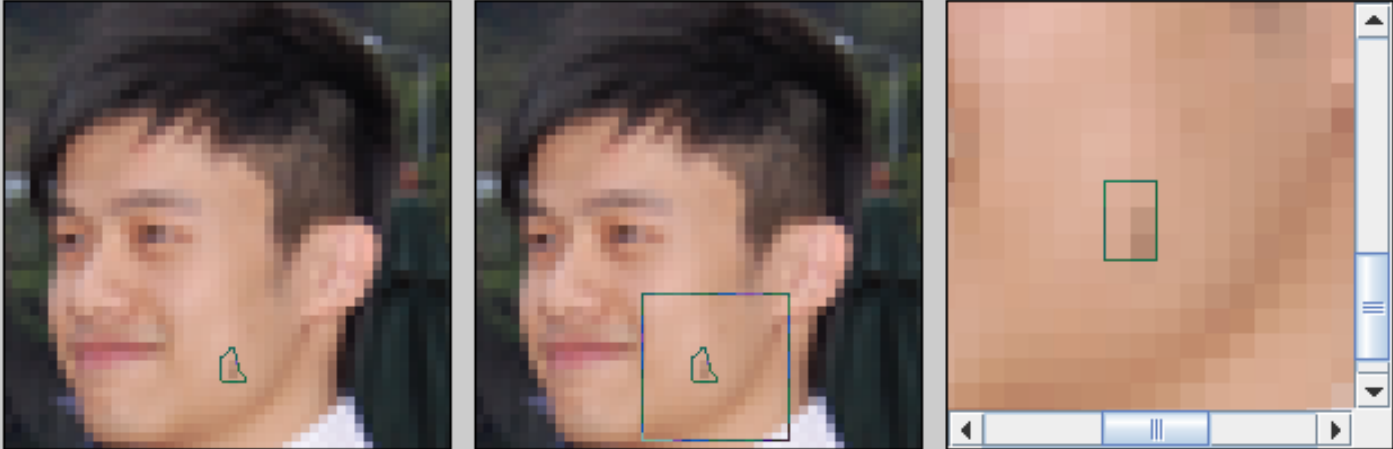
| Time | Type | Pitch | Velocity | Instrument | Hex. Message |
|-------|----------------|-------|----------|------------|--------------|
| 0.00s | Program Change | - | - | 0 Piano | c0 00 |
| 0.00s | Note On | 69 | 127 | - | 90 45 7f |
| 0.27s | Note Off | 69 | 0 | - | 80 45 00 |
| 0.43s | Note On | 71 | 127 | - | 90 47 7f |
| 0.60s | Note Off | 71 | 0 | - | 80 47 00 |
| 0.77s | Note On | 73 | 127 | - | 90 49 7f |
| 0.93s | Note Off | 73 | 0 | - | 80 49 00 |
| 1.23s | Note On | 75 | 127 | - | 90 4b 7f |
| 1.50s | Note Off | 75 | 0 | - | 80 4b 00 |

Playback tempo: 100%

Areas of New Learning Objects - Image

In-Filling Algorithm

Input image Output image Zoomed output image



Picture: **Man with blemish** ▼

Speed: Slowest Fastest

Follow these steps

1. Draw an area on the input image
2. Run the algorithm

Reset

Pseudo-code

```
1. function infill(image)
2.   initialize_2d_array(image)
3.   while(not all 2d_array(x, y) equals to 1)
4.     for each (x, y) on the outermost one pixel boundary of the selected area
5.       color(x, y) = infill_color_average(x, y)
6.     end for
7.   for each (x, y) on the outermost one pixel boundary of the selected area
```

Do this when dragging

- Nothing
- Show polygon line
- Show square line

Areas of New Learning Objects - Video

Video Transition - Barn Door ?

Input Source 1 Input Source 2 Output Video

Picture: Flower Picture: Sunset Settings

Time (frames): 1.20s (36) 0.00s 0.75s 1.50s 2.25s 3.00s ▶ ■

Program Fragment for a Single Frame

```
1. Clear Frame
2. doorSize = width * time / duration
3. Draw Block From Input 1 At (0, 0) With Size ((width - doorSize) / 2, height)
   To Frame At (0, 0) With Size ((width - doorSize) / 2, height)
4. Draw Block From Input 1 At ((width + doorSize) / 2, 0) With Size ((width -
   To Frame At ((width + doorSize) / 2, 0) With Size ((width -
5. Draw Block From Input 2 At ((width - doorSize) / 2, 0) With Size (doorSize,
   To Frame At ((width - doorSize) / 2, 0) With Size (doorSize,
```

▶ Start ▶ Step ▶▶ Run ◻ Stop

Useful?

- Assessment is early stage, on-going
- Early results indicate strong appreciation of the learning objects

Thank you!