

Tablet PC applications for education: database design using Hand-drawn ERD

Dr. Ahmed El-abbassy

El-shorouk Academy, Institute of Computers and Information Technology
P.O Box, El-shorouk, Cairo, Egypt

ahmed_elabbassy@yahoo.com

Abstract: *In the study of Computer science as well as in many other disciplines, Instructors often use hand-drawn diagrams during lectures to illustrate system models, system charts, and database schemas. The use of the Tablet PC as a lecture aid allows presenting lectures with hand-drawn graphics and to annotate and develop a lecture in real time as would be done on a conventional blackboard.*

Instructors are very interested in the use of the Tablet PC to enhance lecture delivery and support active-learning class experiences.

We designed and developed a Tablet PC application that supports database design courses, and assists in drawing entity relationships diagrams (ERD) and generating associated relational database schema, SQL and XML code and data dictionaries.

This paper presents and discusses our experience in the design and implementation of such tablet PC based tools.

Keywords: Tablet PC, Pen Computing, Database Design, Human Computer Interaction (HCI).

1. Introduction

A variety of increasingly popular hardware devices support sketch-based input to computer applications, including the Tablet PC, mobile PDAs, large-screen E-whiteboards, and plug-in tablets for conventional PCs and laptops[1].

Tablet PCs are laptop PCs enhanced with a sensitive screen designed to interact with a stylus [1,2]. They can be used in the same type of setting as the laptop or while standing up, which increases mobility. One can use the stylus as a mouse, draw directly on the screen, or handwrite and use the built-in handwriting recognition feature.

Instructors are very interested in the use of the Tablet PC to enhance lecture delivery and support active-learning class experiences [3,4,5,6]. Some popular Tablet PC applications for instructors are the ink-enabled version of Microsoft Windows Office, Microsoft Windows Journal, OneNote for note taking, Classroom Presenter, and GraphPad for interactive lecture delivery [7].

Much recent research in HCI and user interfaces has demonstrated the potential of such sketching-based user interfaces to enhance the efficiency and effectiveness of user interfaces, particularly for lecture delivery. There is a significant market for the development of Tablet PC tools because of the Tablet PC drawing, handwriting and mobility capabilities.

In order to introduce the Tablet PC technology as a research area in our institute and to encourage the development of Tablet PC applications as lecture aid, we decided to develop a Tablet PC tool called ERDRAW with the aim to support the database design process.

2. Motivation

Instructors of Database courses should be supported by software tools to demonstrate to student the various steps of Database Design.

There is a significant market of CASE tools that can be used to support the teaching of Database design courses. However, most existing software tools are drag-and-drop based, and lack the support for sketching-based input [8]. A small number of tools have attempted to provide sketching-based UML and user interface design support [9,10,11,12,13,14].

In order to design and implement ERDRAW application, we identified the following key requirements:

- Full Database life cycle support.
- Flexible recognition and conversion. Diagramming tool users need flexible control over when content is recognised and converted.
- Recognition accuracy: Accurate shape and text recognition has been shown to be essential for sketch interfaces [15]. However, users should be provided with the ability to very easily over-ride the recogniser when it makes an error.
- Seamless movement: Users need to be able to move easily between sketching-based diagram input and annotation and conventional mouse-driven editing of diagram content.

3. Related Work

Many CASE tools support Database modelling. Many are using conventional mouse/keyboard input and formalised icons.

A small number of tools have attempted to provide sketching-based Database design support such as TabletERD [16]. TabletERD is an open source CASE tool based on Tablet PC technology. The user can draw ERD components: entity types (rectangle) and relationship types (line between entity and/or relationship types). When a relationship type is recognized, a diamond is added to the ERD. Attributes belong to entity and relationship types; attribute properties such as names and types are added from drop-down menus.

4. Our Approach

Database design focuses on the specification of the database schema. It is based on the design of an Entity Relational diagram (ERD) [17]. EDR diagram is a high-level graphical view of the data and how they are related. An ERD is pictured as a set of entity types (with their roles), relationship types (with their cardinalities), and attributes of entity and relationship types (including key attributes). Entity types are represented by rectangles and relationship types by diamonds. ERDs are then transformed into a set of relations and SQL tables. Additionally, the normalization theory provides a mechanism for analyzing and refining the relational schema produced by an ERD design.

ERDRAW is architected to support the major steps of database design process which is illustrated in figure (1) [17].

Step 1: Conceptual design

Create a conceptual schema for the database using high level data model (ER model).

Step 2: Logical design

The actual implementation of the database, using the relational database model.

The ER-to-Relational Mapping Algorithm has six main steps:

- Step 1: Mapping of Regular Entity Types
- Step 2: Mapping of Weak Entity Types
- Step 3: Mapping of Binary 1:1 Relation Types
- Step 4: Mapping of Binary 1:N Relationship Type:
- Step 5: Mapping of Binary M:N Relationship Type:
- Step 6: Mapping of Multivalued attributes.

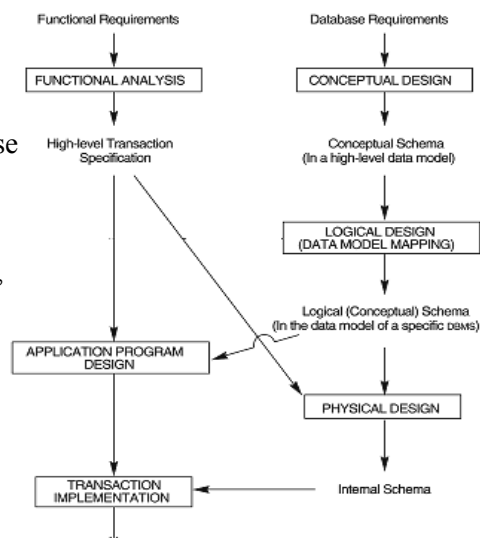


Figure 1: The Main Phases of Database Design (a Simplified Diagram)

With ERDRAW, the user can draw ERD components: entity types (rectangle), relationship types (diamond), and attributes (ovals) that belong to entity and relationship types; entity, relationship and attribute properties such as names and types are added from drop-down menus. ERDRAW generates XML and SQL code for various database management systems including MySQL, Oracle, IBM DB2 and Microsoft SQL Server. ERDs can be saved as JPEG images.

Figure 2 illustrates the basic architecture of ERDRAW.

As shown in figure (2), ERDraw completes its task in three main processes:

Process 1: Ink recognition & Analysis

It incorporates: ink capturing, text and shapes recognition and ink analysis to understand the model semantic and to find out the relationships between the model elements.

The results of recognition are then validated, and the outputs of that phase are produced and saved in three files:

- Ink file: to save the hand-drawn ERD.
- JPEG file: An image of the recognized ERD.
- XML file: to save the model Semantic (ERD elements with

its properties and relationships between elements)

Process 2: Database Creation

It incorporates the following: Transforming the ERD expressed in XML format into relational database schema and generating the corresponding SQL script, then executing the scrip in order to create the database structure. Finally ERDaw

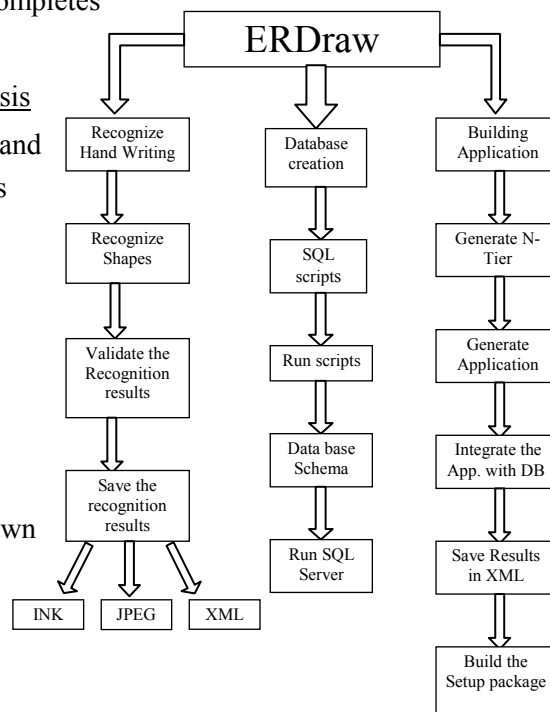


Figure 2: Basic Architecture of ERDraw

activates SQL server in order to obtain the diagram of the relational database schema. Database schema is also saved in XML format.

Process 3: Building a Desktop Database application

The task of ERDaw continues by building a comprehensive desk top database application to interact with the created database. The generated application provides a GUI interface and supports for basic operations such as Insert, Update, Delete and Select

5. Implementation

ERDraw is developed in C#. Ink manipulation is supported by Microsoft INK library

The main challenges encountered during the development of ERDraw include:

5.1 Capturing of Digital Ink

Digital ink is organized into Ink Overlays, Strokes, and Packet data as illustrated in Figure 3 [18]. A Stroke object consists of packet information (x, y,...) that represents the Ink at a certain point on the coordinate system, which starts 0,0 at the upper-left corner.

the Ink Overlay class, grant access to the native ink information through an Ink object. This object encapsulates the actual ink data and ways to manipulate the ink. One of the most commonly used members of the ink object is the Strokes collection. This collection contains one item for each stroke the ink object contains.

Every time the user touches the display with the pen to, draws something, and lifts the pen back up a new stroke is added to the collection.

Using Microsoft Tablet PC SDK, it is simple to develop ink-enabling applications. The InkOverlay class supports different interactions with Ink: Ink (add strokes), Select, Delete, and Change Ink color.

We were interested in manipulating gestures to erase ink using Tablet-specific feature “end-of-pen” eraser [19]. This allows the user to turn the pen around and use the end to erase ink. With this procedure, our application can recognize a scratch out gesture to erase a section of previously written Ink. Other commonly

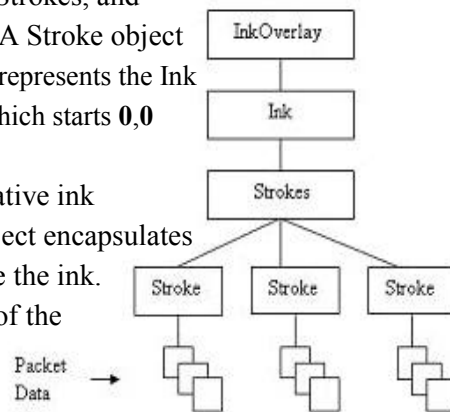


Figure 3: Object Model for ink

performed operations are loading and saving ink. Ink is saved in files with ink serialized format.

5.2 Recognition of Handwriting

The real power of Tablet PCs comes from the ability to analyze and recognize digital Ink [20]. Recognition of handwriting is important as it allows for the conversion of digital Ink into standard text strings. Almost all Tablet PC operating systems ship with multiple recognizers including an English handwriting recognizer.

The collection of strokes is passed to the recognizer engine which returns the most likely recognition result (with confidence level “strong”). Alternate recognition results (with confidence level “poor”) can also be accessed.

We are interested in advanced recognition features that can be used to improve recognition quality and to recognize text that grows to multiple lines such as factoids, baseline information, and the “Ink Divider” [20]. Factoids allow indicating to the recognizer the text pattern that is expected to be found.

Lines of text are the equivalent of baselines. The baseline is crucial in all handwriting recognition because it tells the recognizer where letters start and end.

“Ink Divider” can take a collection of strokes and apply logical divisions. This way we can identify paragraphs, lines, and words.

5.3 Recognition of Shapes and relationship between shapes

Digital Ink is only a collection of lines rendered on the screen, but with Ink recognition and Ink analysis, that information can be turned into meaningful information. Such information can be in the form of text and drawings. In the case of gestures, it can be commands, and in the case of spatial analysis, it can be contextual information such as the relationship between two shapes.

Ink analysis provides a more consistent approach with spatial analysis, context parsing, and handwriting recognition; all provided by a single service.

We used the Ink analysis service to recognize shapes (ovals, lines, rectangles, diamonds, and more). The ink position property is used to correlate shapes with text.

5.4 Recognition efficiency

If large amount of data is to be recognized, we may think about asynchronous Ink recognition. In asynchronous scenarios, the recognizer context is created and stored away for later use. It is also assigned a stroke collection immediately. Everything else is handled through an event model. Recognition happens on a secondary thread, but all events are funneled back to the UI thread (“your” thread). There are two events of importance: Whenever a new stroke is added to the Ink collector, this stroke also has to be added to the recognizer context and subsequently, background

recognition has to be triggered, which will cause a recognition event on the recognizer context object.

6. Usage Example

We now discuss some typical usage pattern

1. Draw and Edit ERD

As illustrated in Figure (4), the first step is to capture the hand-drawn ERD. The user will be able to edit ink (Shapes, Text handwriting) to draw his/her ERD. The drawing panel has a grid that divides the drawing area into small squares to enhance the handwriting. According to their preference, users can decide to start with an empty diagram or to load an ERD partially prepared in a previous session. The user has a range of actions to edit Ink: add ink, delete ink, select ink, and change ink color and width. The hand-drawn ERD is finally saved in a file with ink serialization format for further processing.

During the editing session, ink is asynchronously recognized. Every time the user adds or deletes strokes, recognition results are updated and displayed on the screen. The results are displayed in terms of recognized shapes and text. This methodology is important to produce hand-drawn ERD with reduced number of recognition errors.

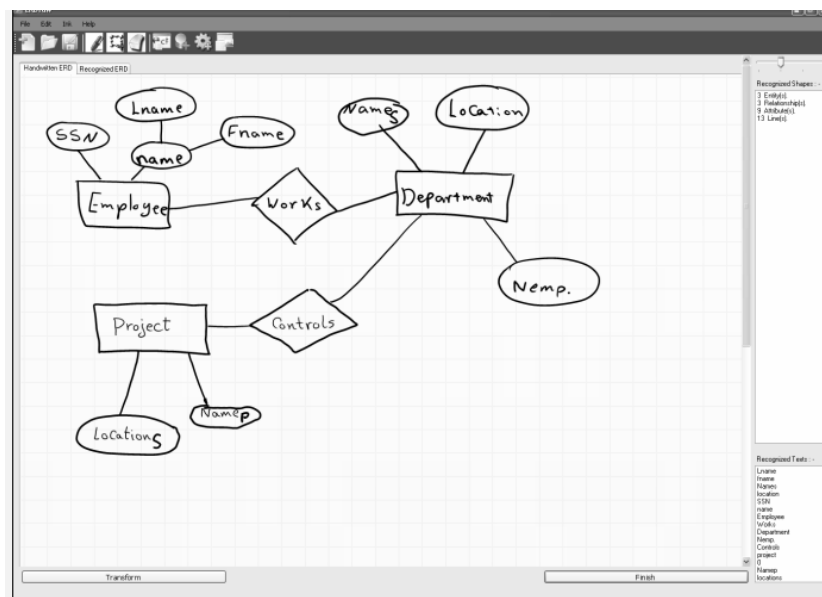


Figure 4: Screen for editing the ink

2. Finish ERD editing

As illustrated in Figure (5), the user indicates the end of editing session by pressing the button “Finish”. ERDraw analyzes the recognition results to check the correctness of the recognized ERD. If the operation is successful, the ERD specifications are complemented by capturing the properties of ERD elements: attributes, entities, and relationships.

The screenshot shows a software window titled "Completion Form" for "TABLET ERD". It contains three main sections for configuring ERD elements:

- Attribute List:** Includes fields for Attribute Name, Attribute Type, Attribute DataType, and Attribute Length. There is also a checkbox for "Candidate Key" and an "Apply" button.
- Entity List:** Includes fields for Entity Name and Entity Type, with an "Apply" button.
- Relationship List:** Includes fields for Relationship Name, Relationship Type, Relationship Cardinality, and Participation, with an "Apply" button.

An "OK" button is located at the bottom right of the window.

Figure 5: Form to insert properties

3- Draw a recognized ERD

The next step as illustrated in Figure 6 is to transform the recognized ERD into image format.

Tablet PC applications for education: database design using Hand-drawn ERD

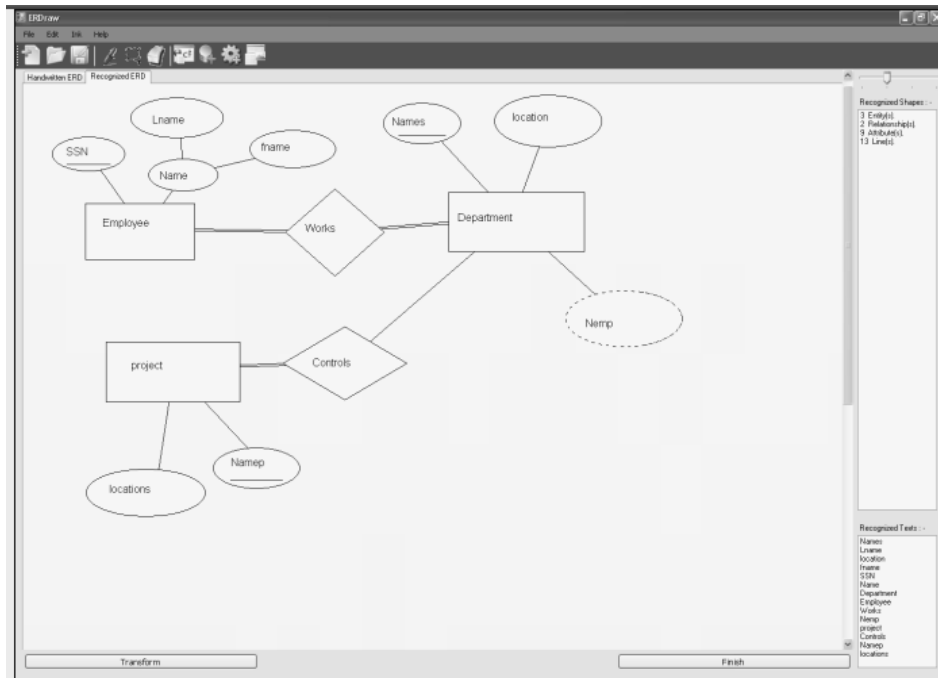


Figure 6: ERD in JPEG format

4. Transform ERD into XML

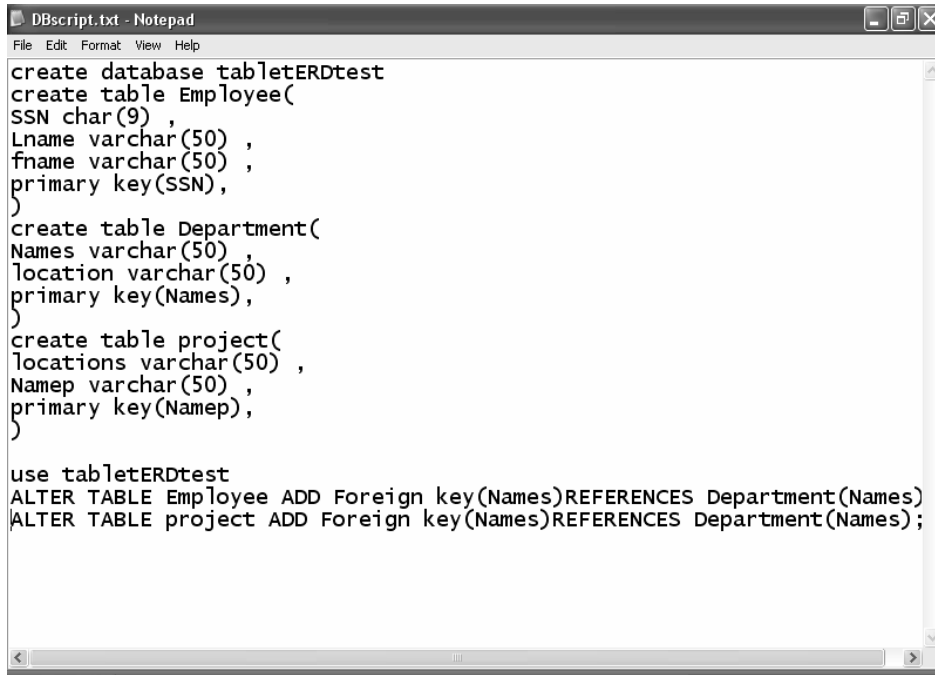
Once the ERD is successfully recognized, it is transformed into XML format for further processing.

5. Transform ERD into Relational Schema

In this step algorithms are applied to transform the ERD expressed in XML format into relational database schema in terms of tables and constraints.

6. Generate SQL script and XML database schema

As illustrated in Figure (7), the results of ER- to-Relational Mapping is a SQL script



```
create database tableERDtest
create table Employee(
SSN char(9) ,
Lname varchar(50) ,
fname varchar(50) ,
primary key(SSN),
)
create table Department(
Names varchar(50) ,
location varchar(50) ,
primary key(Names),
)
create table project(
locations varchar(50) ,
Namep varchar(50) ,
primary key(Namep),
)

use tableERDtest
ALTER TABLE Employee ADD Foreign key(Names)REFERENCES Department(Names)
ALTER TABLE project ADD Foreign key(Names)REFERENCES Department(Names);
```

Figure 7: SQL Script

7. Create Database and database schema diagram.

The produced SQL script is submitted to a relational DBMS in order to create the database structure and to output the database schema as illustrated in Figure (8).

Tablet PC applications for education: database design using Hand-drawn ERD

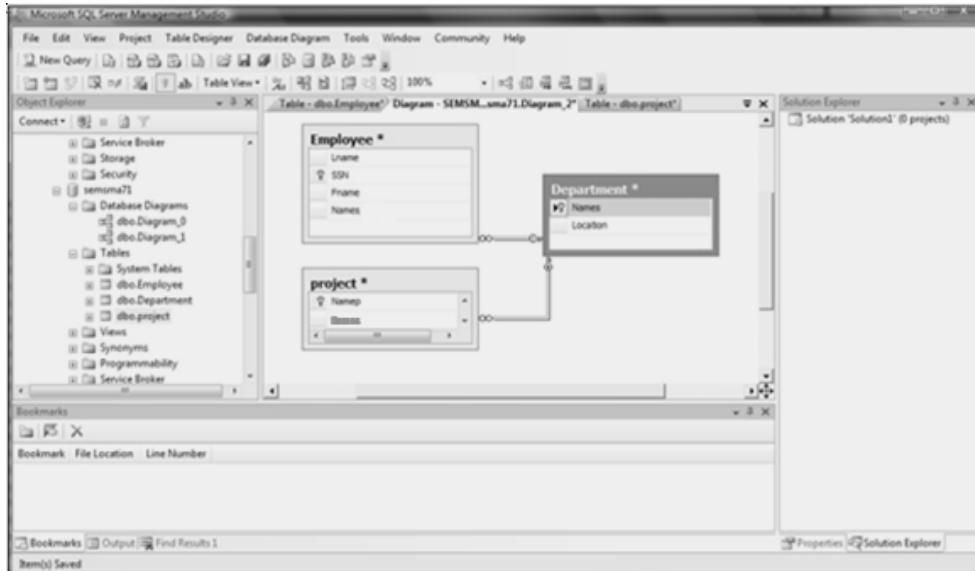


Figure 8: Relational Database Schema

8. Build data base application

ERDraw incorporates a functionality that allows generating a GUI application to access the created database for basic operations such as; Select, Insert, Delete, and Update.

As illustrated in Figure (9), the user can activate this application to do basic operations on the created database.

Figure 9 : Application Form to interact with the Database

7. Conclusion

ERDraw can be used and evaluated in a database design course. We expect to focus our evaluation on student opinion and results of use in the course. We will document the ease and effectiveness of using ERDraw to assist in learning of database design. The drawing and the code generation features of ERDraw will be emphasized in the evaluation. We plan to extend the features of ERDraw by capturing the properties of ERD elements by handwriting (instead of drop down menus) and introducing best practice design tips for the user based on the normalization theory.

References

- [1] Steven J. Timmins, "Tablet PC: Blackboard to the Web", SIGUCCS'04, ACM, October 10-13, 2004, Baltimore, Maryland, USA 2004.
- [2] Cheryl L. Willis, Susan L. Miertschin, "Tablet PC's as Instructional Tools or the Pen is Mightier than the 'Board! ", SIGITE'04, October 28-30, 2004, Salt Lake City, Utah, USA, ACM 2004.
- [3] Simon, B., Anderson, R., Hoyer, C., and Su, J. 2004. "Preliminary experiences with a tablet PC based system to support active learning in computer science courses". In Proceedings of the 9th Annual SIGCSE Conference on innovation and Technology in Computer Science Education (Leeds, United Kingdom, June 28 - 30, 2004). ITiCSE '04. ACM Press, 2004.
- [4] Gardiner, L. "Why We Must Change: The Research Evidence". The NEA Higher Education Journal. 16 (1998): 71-88 [On-line]. Available: <http://www.nea.org/he/heta98/s98pg71.pdf>.
- [5] Bonwell, C., Eison, J. "Active Learning: Creating Excitement in the Classroom". ASHE-ERIC Higher Education Report No. 1. Washington, D.C., 1991.
- [6] Sam Kamin Michael Hines, "A System for Developing Tablet PC Applications for Education", SIGCSE'08, March 12-15, 2008, Portland, Oregon, USA, Copyright 2008 ACM
- [7] Roy P. Pargas, Samuel Bryfczynski, "Using Ink to Expose Students' Thought Processes", in CS2/CS7, SIGCSE'09, March 3-7, 2009, Chattanooga, Tennessee, USA., Copyright 2009 ACM
- [8] John Grundy, John Hosking, " Supporting generic sketching-based input of diagrams in a domain-specific visual language meta-tool", 29th International Conference on Software Engineering (ICSE'07), IEEE 2007.

- [9] Damm, C.H., K.M. Hansen, and M. Thomsen. "Tool support for cooperative object-oriented design: Gesture based modelling on and electronic whiteboard". Proc Chi 2000.2000: ACM: p. 518-525.
- [10] "Activity Diagrams for Business Process Modelling", Proc HCI 2005 Workshop on Improving and Assessing Pen-based Input Techniques, Edinburgh, September 2005.
- [11] Chen, Q., Grundy, J.C. and Hosking, J.G. "An E-whiteboard Application to Support Early Design-Stage Sketching of UML Diagrams", Proc HCC'03, Auckland, October 2003, 219-226.
- [12] Landay, J. and B. Myers. "Interactive sketching for the early stages of user interface design". Proceeding of Chi '95 Mosaic of Creativity. 1995. ACM: p. 43-50.
- [13] Plimmer, B.E. and M. Apperley, "Software for Students to Sketch Interface Designs". Proc Interact. 2003. p. 73-80.
- [14] Tracy Hammond, Brian Eoff, "Free-Sketch Recognition: Putting the CHI in Sketching", CHI 2008 Proceedings, April 5-10, 2008 · Florence, Italy.
- [15] MacKenzie, I. S., & Zhang, S. The immediate usability of Graffiti. Proceedings of Graphics Interface '97, pp. 129-137,1997.
- [16] Sokharith Sok, Christelle Scharff, "Database Design With TabletERD, 36th ASEE/IEEE Frontiers in Education Conference, October 28 – 31, 2006. San Diego, CA.
- [17] Elmasri , "Fundamentals Of Database Systems", 5/E, ISBN 0-321-41506-X.Publication: Addison Wesley, July 2007.
- [18] Frankgo, "Understanding Strokes and Digital Ink", Mar 2007, Code Project
- [19] Markus Egger, "Creating Tablet PC Applications with VS.NET, Code" Magazine, Article source: CoDe (2003- September/October).
- [20] Markus Egger, "Ink Recognition and Ink Analysis, Code" Magazine, Article source: CoDe (2005 - Vol. 3 - Issue 1 - Tablet PC and Mobile PC)A