# บทที่ 3
# วิธีดำเนินการศึกษา

การศึกษาครั้งนี้มีวัตถุประสงค์เพื่อพัฒนาระบบการประยุกต์ใช้การค้นคืนสารสนเทศบนฐานข้อมูลเชิงสัมพันธ์โดยมีรายละเอียดในการดำเนินการดังต่อไปนี้

## 3.1 การเตรียมความพร้อม โดยมีวิธีการดังต่อไปนี้

3.1.1 ติดตั้งฐานข้อมูลเชิงสัมพันธ์ เช่น MSSQL เป็นต้น

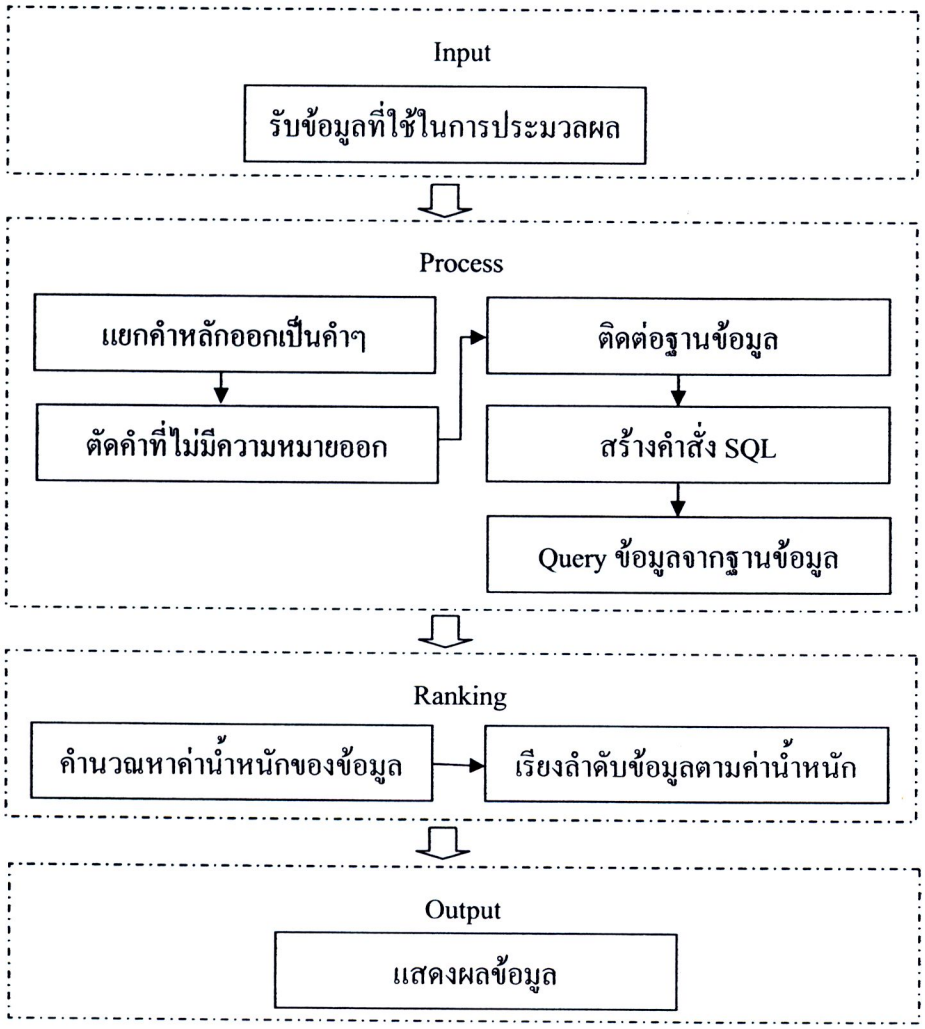
3.1.2 ติดตั้งโปรแกรม .Net Framework Version 2.0 ขึ้นไป

3.1.3 ติดตั้งโปรแกรมสำหรับพัฒนาระบบ เช่น Microsoft Visual Studio เป็นต้น

3.1.4 ติดตั้งโปรแกรม Web Server เช่น IIS เป็นต้น

## 3.2 หลักการทำงานของ Algorithm

หลักการทำงานของ Algorithm การประยุกต์ใช้การค้นคืนสารสนเทศบนฐานข้อมูลเชิงสัมพันธ์จะแบ่งออกเป็น 4 ขั้นตอนแสดงดังภาพ 3.1




**ภาพ 3.1** แสดงกระบวนการทำงานของ Algorithm

3.2.1 Input เป็นส่วนที่รับข้อมูลต่างๆ ที่เกี่ยวข้องหรือต้องการใช้ในการประมวลผลข้อมูลจำประกอบไปด้วยข้อมูลนี้

3.2.1.1 คอลัมน์ที่ต้องการใช้ประมวลผลข้อมูล

3.2.1.2 คอลัมน์ที่ไม่ต้องการให้ Algorithm ประมวลผล

3.2.1.3 ตารางที่ต้องการให้ประมวลผลข้อมูล

3.2.1.4 คำหลักที่จะใช้ในการประมวลผลข้อมูล

3.2.1.5 ค่าที่ใช้ในการติดต่อระบบฐานข้อมูลเชิงสัมพันธ์

การรับข้อมูลต่างๆ ที่ใช้ในการประมวลผลของ Algorithm จะรับผ่านทาง Constructor Method ของ Class Information Retrieval ดังตัวอย่าง


```
constructor public InformationRetrieval(
        string[] dataField,
        string[] fieldCondition,
        string[]dataTable,
        string dataKeyword,
        string connectStringName
)
```

3.2.2 Process เป็นขั้นตอนที่ใช้ในการประมวลผลข้อมูลต่างๆ ที่รับมาจากขั้นตอน Input และทำหน้าที่ในการติดต่อระบบฐานข้อมูลเชิงสัมพันธ์ โดยจะแบ่งกระบวนการทำงานเป็นขั้นตอนได้ดังนี้

3.2.2.1 แยกคำหลักออกเป็นคำๆ คือ จะนำคำหลักที่รับเข้ามาจากขั้นตอน Input มาแยกออกเป็นคำๆ ทำการประมวลผลโดย Method ชื่อว่า textSpit() ดังตัวอย่าง

```
private List<string> textSpitWord(string text) {
        var w = new List<string>();
        string thisText = this.badWord(text);
        using ( BreakIterator bi =
BreakIterator.CreateWordInstance(Locale.GetUS())) {
            bi.SetText(thisText);
            int start = bi.First(), end = bi.Next();
            while (end != BreakIterator.DONE) {
                w.Add(thisText.Substring(start, end - start));
                start = end;
                end = bi.Next();
            }
        }
        return w;
}
```

**หมายเหตุ :**

- Locale.GetUS() หมายถึง รูปแบบที่ใช้จะต้องเป็นภาษาอังกฤษหรือภาษาไทย หากเป็นภาษาอื่นต้องเปลี่ยนเป็นอย่างอื่นตามที่ ICU กำหนด Algorithm นี้จะเรียกใช้ความสามารถของ Alogithm ICU ที่บริษัท IBM เป็นผู้พัฒนาอีกทีหนึ่ง
- Code โปรแกรมชุดนี้อ้างอิงมาจากเว็บไซต์ http://code.google.com/p/icu4net/ เมื่อวันที่18 ธันวาคม 2555

3.2.2.2 ตัดคำที่ไม่มีความหมายออก คือ จะนำคำที่ได้จากการแยกคำหลักของขั้นตอนที่ 3.3.2.1 มาประมวลผลว่าคำเหล่านั้นเป็นคำที่มีความหมายในการค้นหาข้อมูลหรือไม่ หากไม่มีก็ให้ลบออกจากคำหลักที่ใช้ในการค้นหาข้อมูล เช่นคำว่า is a the เป็นต้น จะประมวลผลโดย Method ที่ชื่อว่า badWord() ดังตัวอย่าง

```
protected string badWord(string w) {
    return w.Replace(" to ", " ")
            .Replace(" is ", " ")
            .Replace(" of ", " ")
            .Replace(" the ", " ")
            .Replace(" he "," ")
            .Replace(" she "," ")
            .Replace(" it "," ");
}
```

3.2.2.3 ติดต่อฐานข้อมูล นำค่าที่ใช้ในการติดต่อระบบฐานข้อมูลเชิงสัมพันธ์ ที่รับมากจากขั้นตอน Input มาทำการเชื่อมต่อระบบฐานข้อมูลเชิงสัมพันธ์ ดังตัวอย่าง

```
SqlConnection objConn = new SqlConnection(connectStringName);
```

**หมายเหตุ :**

- รูปแบบที่ใช้เป็นตัวอย่างรูปแบบการติดต่อฐานข้อมูล MSSQL
- connectStringName คือชื่อ Parameter ที่ Constructor Method ที่ของ Class InformationRetrieval รับเข้ามา

3.2.2.4 สร้างคำสั่ง SQL คือ นำเอา Parameter ที่รับเข้ามาจากขั้นตอน Input และ คำต่างๆ ที่ผ่านการประมวลผลมาจากขั้นตอน Process ในหัวข้อที่ 3.3.2.1 และ 3.3.2.2 มาเป็นสร้างคำสั่ง SQL ตามรูปแบบมาตรฐานของระบบฐานข้อมูลเชิงสัมพันธ์ โดย Parameter ที่รับมาจากขั้นตอน Input จะประกอบไปด้วย Parameter ดังต่อไปนี้ dataField, fieldCondition และ dataTable จะประมวลผลโดย Constructor Method ของ Class InformationRetrieval ดังตัวอย่าง

```
bool dataOrFirstLoop = true;
for (int i = 0; i < dataField.Length; i++) {
    if (!string.IsNullOrEmpty(dataField[i])) {
        if (!dataOrFirstLoop) {
            tempField += " + ' ' + ";
        }
        tempField +=
            "(CASE WHEN " + dataField[i] + " IS NULL
            THEN " ELSE CONVERT(varchar(MAX), " + dataField[i] +
")
            END)";
    }
    dataOrFirstLoop = false;
}
bool dataFieldFirstLoop = true;
for (int i = 0; i < dataField.Length; i++) {
    if (!string.IsNullOrEmpty(dataField[i])) {
        if (!dataFieldFirstLoop) { tempNormalField += ","; }
        tempNormalField += dataField[i];
    }
   dataFieldFirstLoop = false;
}
bool tempstatusConditionFirstLoop = true;
```

```
for (int i = 0; i < statusCondition.Length; i++) {
    if (!string.IsNullOrEmpty(statusCondition[i])) {
        if (tempstatusConditionFirstLoop) {
            tempstatusCondition +=
this.replaceTextSpacialChar(statusCondition[i]);
            tempstatusConditionFirstLoop = false;
            } else {
                tempstatusCondition += " AND " +
                this.replaceTextSpacialChar(statusCondition[i]);
            }
        }
    }
    bool firstLoop = true;
    string[] _keyTemp = new string[100]; // Set keyword for DB
    string[] _argTemp = new string[100]; // Set parametor for DB
    int keyTempIndex = 0;
    string conditionOrTMP = "";
    foreach (string show in textSpit(keyword)) {
        if (!string.IsNullOrEmpty(show) && show != " ") {
            // Set new parametor name for DB
            _argTemp[keyTempIndex] = "@keyword" +
keyTempIndex.ToString();
            // Set new keyword value for DB
            _keyTemp[keyTempIndex] = show;
            if (!firstLoop) {
                conditionOrTMP += " OR ";
                conditionOrTMP += tempField + " LIKE '%'+" +
                _argTemp[keyTempIndex] + "+'%' ";
                firstLoop = false;
                keyTempIndex++;
```

```
            }
        }
    }
    if (!string.IsNullOrEmpty(tempstatusCondition)) {
        _sql =
            "SELECT TOP(" + maxRowsForDisplayDataResult + ") " +
            tempFieldData + " AS data, " + tempNormalField + " FROM "
+
            dataTable + " WHERE (" + conditionOrTMP + ") AND (" +
            tempstatusCondition + ")";
    } else {
        _sql =
            "SELECT TOP(" + maxRowsForDisplayDataResult + ") " +
            tempField + " AS data, " + tempNormalField + " FROM " +
            dataTable + " WHERE (" + conditionOrTMP + ")";
    }
```

3.2.2.5 Query ข้อมูลจากฐานข้อมูล คือกระบวนการที่นำเอาคำสั่ง SQL ที่ได้จากการสร้างจากหัวข้อที่ 3.3.2.4 ส่งไปยังระบบจัดการฐานข้อมูลเชิงสัมพันธ์ เพื่อร้องขอข้อมูลที่ต้องการ จะประมวลผลโดย Constructor Method ของ Class Information Retrieval ดังตัวอย่าง

```
StringBuilder objSB = new StringBuilder();
using (SqlConnection objConn = new SqlConnection(
WebConfigurationManager.ConnectionStrings[connectStringName].Conn
ectionString)) {
    objConn.Open();
    SqlCommand objCM = new SqlCommand();
    objCM.CommandText = _sql;
    objCM.CommandType = CommandType.Text;
    objCM.Connection = objConn;
  for (int i = 0; i <= _argTemp.Length; i++) {
```

```
        if (!string.IsNullOrEmpty(_argTemp[i])) {
            objCM.Parameters.AddWithValue(
                _argTemp[i].ToString(),
                _keyTemp[i].ToString());
        } else { break; }
}
SqlDataReader objDR = objCM.ExecuteReader();
bool loopFirst = true;
string[] attField = tempNormalField.Split(',');
string fieldNameDeleteTablePrefix = "";
while (objDR.Read()) {
    if (loopFirst) {
        objSB.Append(objDR["data"].ToString());
        for (int i = 0; i < attField.Length; i++) {
            fieldNameDeleteTablePrefix = attField[i].Trim().Substring(
            attField[i].Trim().IndexOf('.') + 1);
            objSB.Append(
                sysbolForSplitAttibute + attField[i] +
                sysbolForSplitAttibuteAndValue +
                objDR[fieldNameDeleteTablePrefix].ToString());
        }
        loopFirst = false;
    } else {
        objSB.Append(sysbolForSplitTupleTree +
objDR["Data"].ToString());
        for (int i = 0; i < attField.Length; i++) {
            fieldNameDeleteTablePrefix =
attField[i].Trim().Substring(
            attField[i].Trim().IndexOf('.') + 1);
            objSB.Append(
```

```
                    sysbolForSplitAttibute + attField[i] +
                    sysbolForSplitAttibuteAndValue +
                    objDR[fieldNameDeleteTablePrefix].ToString());
            }
        }
    }
    objDR.Close();
}
```

**หมายเหตุ :** รูปแบบที่ใช้เป็นตัวอย่างรูปแบบการติดต่อฐานของมูล MSSQL

3.2.3 Ranking

3.2.3.1 คำนวณหาค่าน้ำหนักของข้อมูล คือกระบวนการคำนวณหาค่านำหนักของข้อมูลที่ได้จากการประมวลผลตามหัวข้อที่ 3.3.2.5 โดยจะใช้หลักการคำนวณของ การค้นคืนสารสนเทศ ดังต่อไปนี้

$\mathbf{W_{ij} = IDF_i * TF_{ij}}$

$\mathbf{IDF_i = \log_2(n / DF_i)}$

**โดยที่**

$W_{ij}$ คือ ค่าน้ำหนักของแต่ละคำที่ค้นหาเจอ

$TF_{ij}$ คือ จำนวนความถี่ของแต่ละคำ ที่ค้นหาเจอในแต่ละ Term

$DF_i$ คือ จำนวนความถี่ของคำทั้งหมดที่ปรากฏในทุกๆ เอกสาร

$IDF_i$ คือ ค่าที่ได้จากการคำนวณ $Log_2$ ของจำนวนคำทั้งหมดที่ค้นพบหารด้วยจำนวนความถี่ของคำทั้งหมดที่ปรากฏในทุกๆ เอกสาร

n คือ จำนวนคำทั้งหมดที่ปรากฏในทุกๆ Term รวมกัน

จะประมวลผลโดย Method ที่ชื่อว่า SeparateWordsDataBind() ดังตัวอย่าง

```
public DataTable SeparateWordsDataBind() {
    try {
        string[] arrTupleTreeSum =
        dataResultAll.ToString().Split(sysbolForSplitTupleTree);
         string[] arrTupleTree = new string[arrTupleTreeSum.Length];
        string[] arrTupleTreeTMP = new string[50];
```

```
        string wordAll = "";
        for (int i = 0; i < arrTupleTreeSum.Length; i++) {
            arrTupleTreeTMP =

arrTupleTreeSum[i].ToString().Split(sysbolForSplitAttibute);
                arrTupleTree[i] = arrTupleTreeTMP[0];
               wordAll += arrTupleTreeTMP[0];
        }
        // จำนวนคำทั้งหมดที่ค้นหาเจอ เฉพาะที่อยู่ในฟิลด์ data
        int n = this.allWordsOfResult(wordAll);
        // จำนวนความถี่ของแต่ละคำ ที่ค้นหาเจอในแต่ละ tuple
        string[,] tf = new string[maxRowsForDisplayDataResult, 2];
        // จำนวนของ  tuple ที่มีคำนั้นปรากฏ
        string[,] df = new string[maxRowsForDisplayDataResult, 2];
         // log ฐาน 2(n/df)
         string[,] idf = new string[maxRowsForDisplayDataResult, 2];
          // weigth = tf * idf
          string[,] w = new string[maxRowsForDisplayDataResult, 2];

         // variable support ---------------------------
         int j;
         bool calDF = true;
         string strDataTMP = "";
         string[] tfTmp = new string[maxRowsForDisplayDataResult];
         string dfTmp = "";
         string idfTmp = "";
         string[] wTmp = new string[maxRowsForDisplayDataResult];
         double[] wSum = new double[maxRowsForDisplayDataResult];
          string[] buffer = new string[arrTupleTree.Length];
         // ----------------------------------------
```

```
// ทำ loop ของเอกสารทั้งหมดที่ค้นเจอ
for (int i = 0; i < arrTupleTree.Length; i++) {
    j = 0;
    strDataTMP = arrTupleTreeSum[i];
    tfTmp[i] = "";
    wTmp[i] = "";
    wSum[i] = 0.00;
    foreach (string show in this.textSpit(keyword)) {
        if (!string.IsNullOrEmpty(show) && show != " ") {
            // Text math this is a math of keyword to show
            strDataTMP = strDataTMP.Replace(
                show,
                "UuuuU" + show + "HuuuH");
            // Set tf value ----------
            // คำที่ใช้ค้นหา
            tf[j, 0] = show;
            //  จำนวนความถี่ของแต่ละคำ  ที่ค้นหาเจอในแต่ละ
tuple
            tf[j, 1] = this.countOccurencesOfWord(
                arrTupleTree[i],
                show.Replace(" ", "")).ToString();
            // ----------

            // Set data to display ----------
            tfTmp[i] += tf[j, 0] + "=" + tf[j, 1] + " ";
            // ----------

            // df ----------
            if (calDF) {
```

```
                    // Set value ----------
                    df[j, 0] = show;
                    // ทำ loop ของเอกสารทั้งหมดที่ค้นเจอ
                    for (int k = 0; k < arrTupleTree.Length; k++) {
                        if (!string.IsNullOrEmpty(df[j, 1])) {
                            if (this.countOccurencesOfWord(
                                arrTupleTree[k], show) > 0) {
                                df[j, 1] =
                                    (Convert.ToInt16(df[j, 1])
+1).ToString();
                            }
                        } else {
                            if (this.countOccurencesOfWord(
                                arrTupleTree[k], show) > 0) {
                                df[j, 1] = "1";
                            } else {
                                df[j, 1] = "0";
                            }
                        }
                    }
                    dfTmp += df[j, 0] + "=" + df[j, 1] + " ";
                }
                // ----------
                // idf ----------
                if (calDF) {

                    idf[j, 0] = show;
                    if (Convert.ToInt32(df[j, 1]) > 0) {
                        idf[j, 1] = (Convert.ToDouble(
                            Math.Log(Convert.ToDouble(n) /
```

```
                    Convert.ToDouble(df[j, 1]), 2))
                ).ToString("###0.00");
            } else {
                idf[j, 1] = "0";
            }
            idfTmp += idf[j, 0] + "=" + idf[j, 1] + " ";
        }
        // ---------------------------------

        // w ---------------------------------
        w[j, 0] = show;
        w[j, 1] = (Convert.ToDouble(
            idf[j, 1]) * Convert.ToDouble(tf[j, 1])
        ).ToString("###0.00");
        wTmp[i] += w[j, 0] + "=" + w[j, 1] + " ";
        wSum[i] += Convert.ToDouble(w[j, 1]);
        // ---------------------------------
    } else {
        tf[j, 0] = "";
        tf[j, 1] = "0";
    }
    j++;
}
calDF = false;
buffer[i] = strDataTMP;
}
// ---------------------------------
// ###################################
// Code อยู่ในหัวข้อ 3.3.3.2
// ###################################
```

```
            // ---------------------------------
        } catch {
            DataTable dt = new DataTable();
            dt = null;
            return dt;
        }
    }
```

3.2.3.2 เรียงลำดับข้อมูลตามค่าน้ำหนัก จะนำเอาค่าน้ำหนักของของแต่ละแถวที่คำนวณได้จากหัวข้อ 3.3.3.1 มาเรียงลำดับในการแสดงผล โดยจะเรียงลำดับจากมากไปหาน้อย จะประมวลผลโดย Method ที่ชื่อว่า SeparateWordsDataBind() ดังตัวอย่าง

```
string[] bufferTMP;
string[] bufferDataTMP;
DataTable dt = new DataTable();
for (int x = 0; x < buffer.Length; x++) {
    bufferTMP = buffer[x].Split(sysbolForSplitAttibute);
    for (int y = 1; y < bufferTMP.Length; y++) {
        bufferDataTMP = bufferTMP[y].Split(
        sysbolForSplitAttibuteAndValue);
        dt.Columns.Add(new DataColumn(
            bufferDataTMP[0].Trim().Substring(
            bufferDataTMP[0].Trim().IndexOf('.') + 1),
            typeof(string))
          );
      }
      break;
}
dt.Columns.Add(new DataColumn("w", typeof(double)));
dt.Columns.Add(new DataColumn("process", typeof(string)));
```

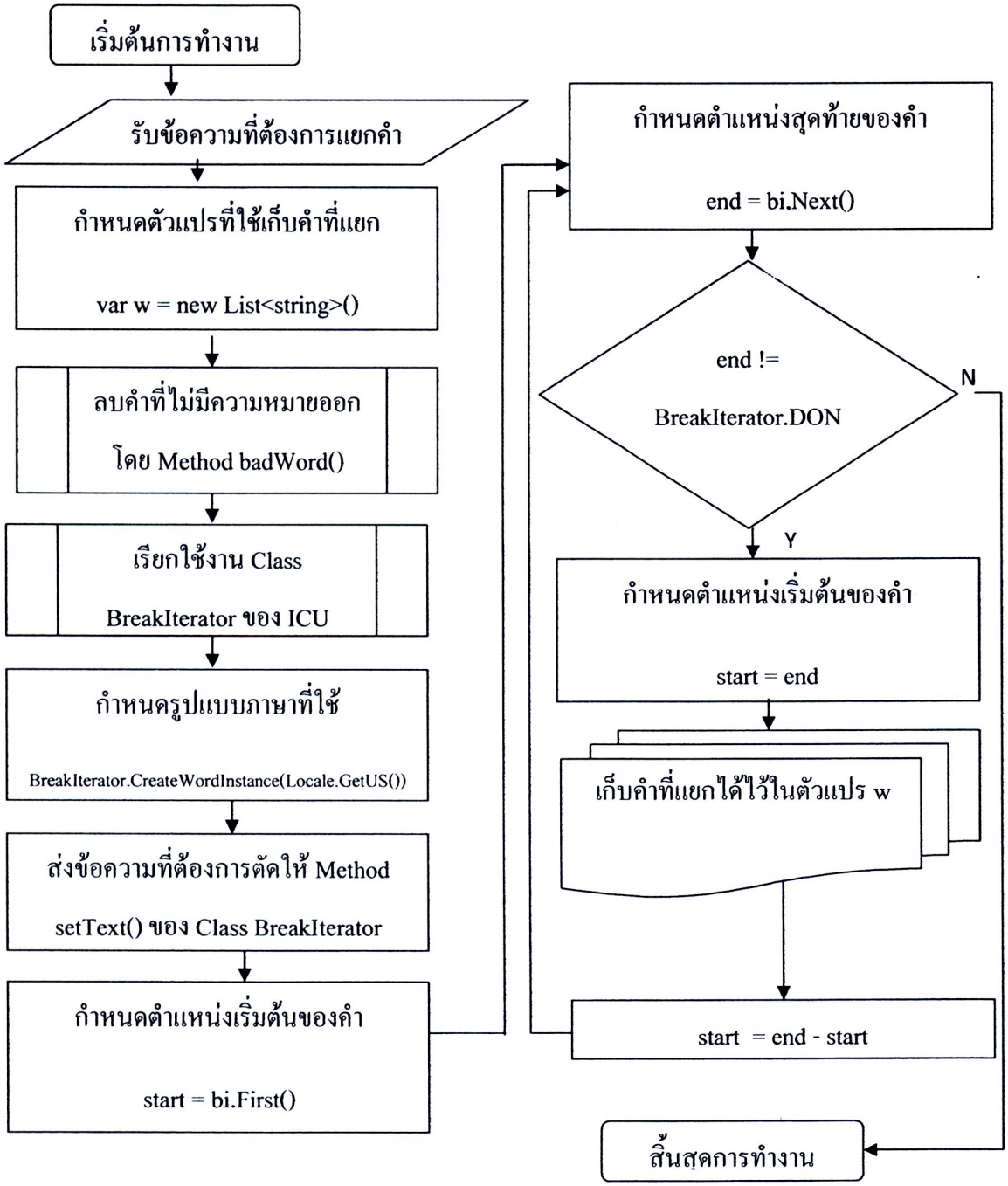
```
for (int x = 0; x < buffer.Length; x++) {
    try {
        DataRow row = dt.NewRow();
        bufferTMP = buffer[x].Split(sysbolForSplitAttibute);
        for (int y = 1; y < bufferTMP.Length; y++) {
            bufferDataTMP = bufferTMP[y].Split(
                sysbolForSplitAttibuteAndValue);
            row[bufferDataTMP[0].Trim().Substring(
            bufferDataTMP[0].Trim().IndexOf('.') + 1)] =
                bufferDataTMP[1].Trim().Replace(
                    "UuuuU", "<u>").Replace("HuuuH", "</u>");
        }
        row["w"] = wSum[x];
        row["process"] = "<b><u>IR Parametor</u></b> : ค่าอื่นที่ใช้ใน
การ
            คำนวน <b>DF</b>{" + dfTmp + "}, <b>IDF[log ฐาน
            2(n/df)]</b>{" + idfTmp + "}, <b>N</b>{" + n + "}";
        dt.Rows.Add(row);
    }
    catch { }
}
dt.DefaultView.Sort = "w DESC";
return dt;
```

3.2.4 Output

3.2.4.1 แสดงผลข้อมูล หลังจากเสร็จสิ้นกระบวนการ Ranking ระบบจะนำเอาข้อมูลที่ผ่านการประมวลผลมาจัดให้อยู่ในรูปแบบของ Dataset เพื่อนำไปแสดงผลข้อมูล

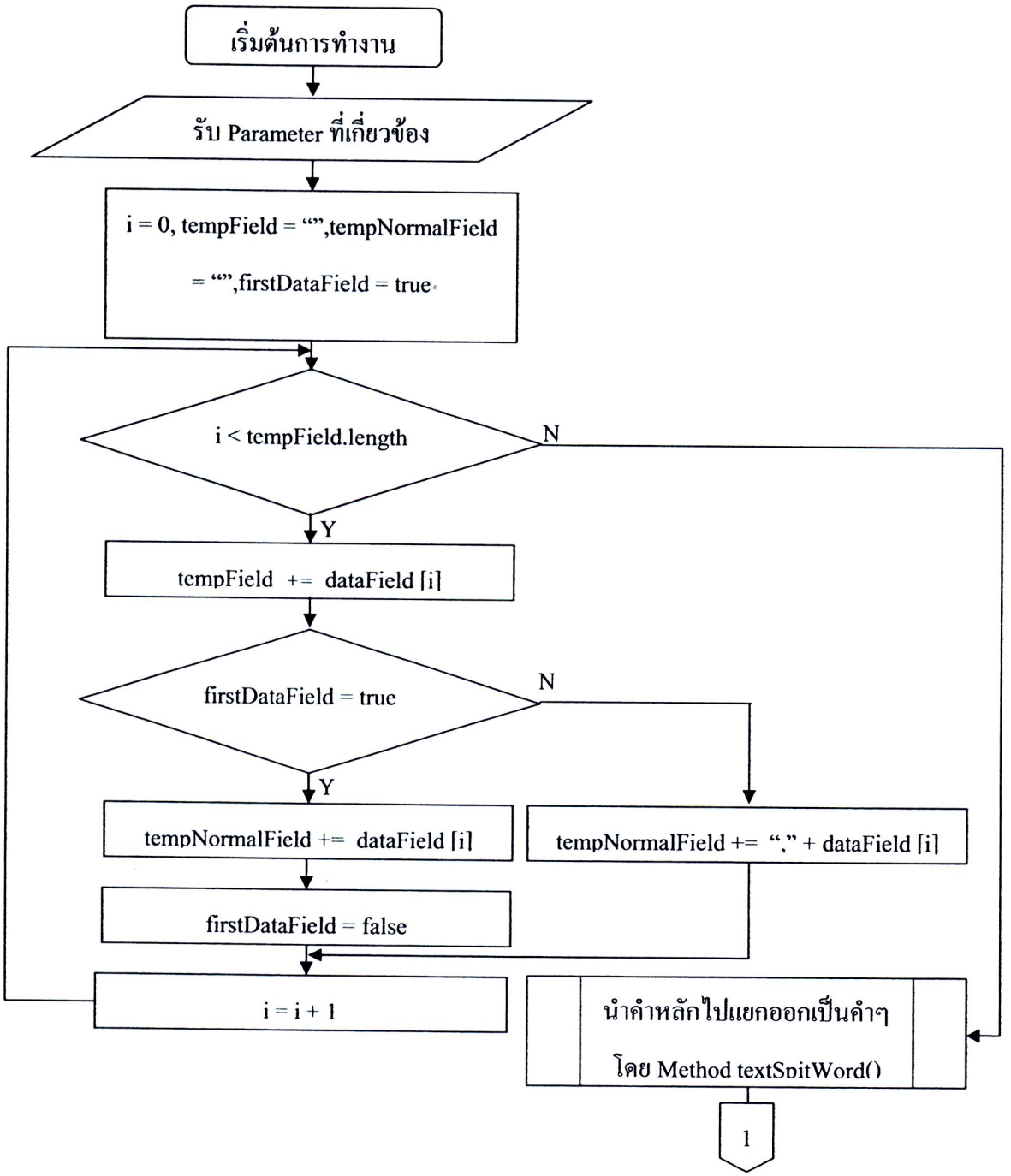
### 3.3 แผงผังการทำงานของ Algorithm

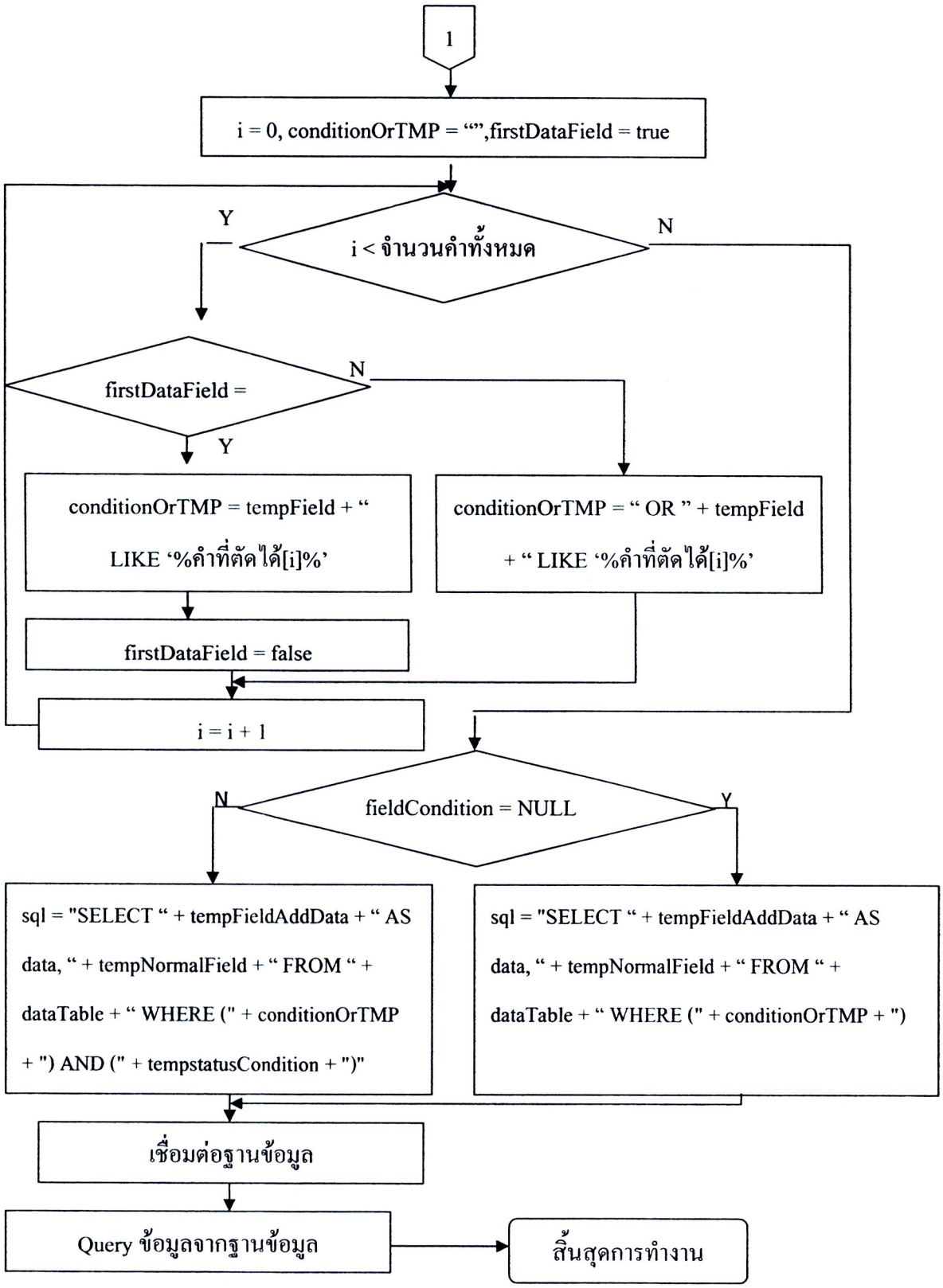
3.3.1 แผงผังการทำงานของ Method textSpitWord() แสดงดังภาพ 3.2




**ภาพ 3.2** แสดงแผนผังการทำงานของ Method textSpitWord()

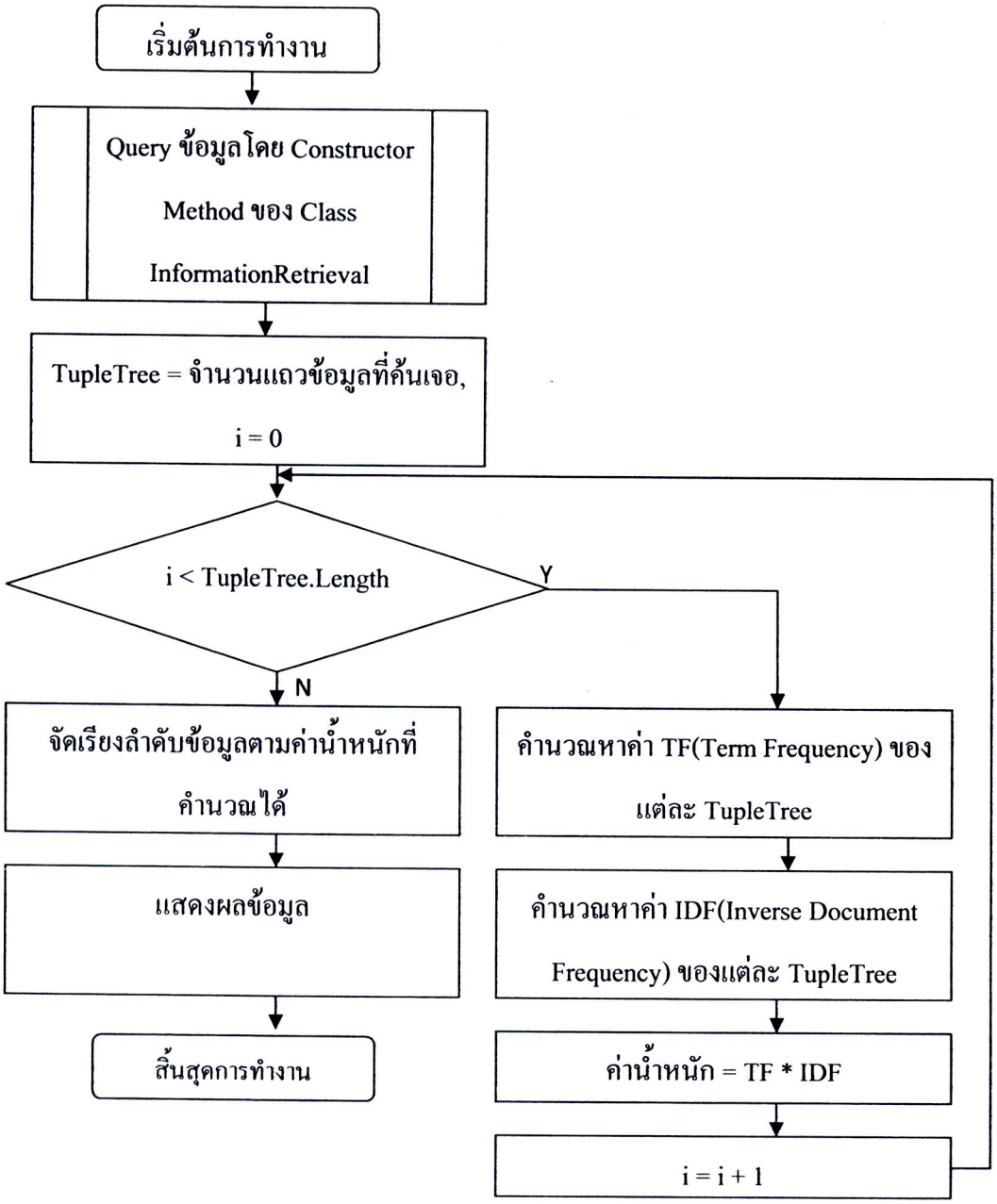
3.3.2 แผนผังการทำงาน Constructor Method ของ Class InformationRetrieval แสดงดังภาพ 3.3






**ภาพ 3.3** แสดงแผนผังการทำงานของ Constructor Method ของ Class InformationRetrieval

3.3.3 แผนผังการทำงานของ Separate Words Data Bind() แสดงดังภาพ 3.4




**ภาพ 3.4** แสดงแผนผังการทำงานของ Method SeparateWords DataBind()

### 3.4 ตัวอย่างการนำไปใช้งาน

การนำเอา Algorithm การประยุกต์ใช้การค้นคืนสารสนเทศบนฐานข้อมูลเชิงสัมพันธ์ไปใช้งาน ผู้วิจัยได้นำเอา Algorithm ไปทดลองใช้งานบนฐานข้อมูลระบบสารสนเทศเพื่อการบริหารงานมหาวิทยาลัยแม่โจ้ ในส่วนของระบบค้นหาข้อมูลงานวิจัย โดยแบ่งขั้นตอนการนำไปใช้งานออกเป็นสองส่วนดังต่อไปนี้

3.4.1 การนำเอา Algorithm ไปประยุกต์เข้ากลับระบบค้นหาข้อมูลงานวิจัย

3.4.1.1 Import ICU4NET เข้า Solution

3.4.1.2 Import class InformationRetrieval() เข้า Solution

3.4.1.3 นำเอา Class InformationRetrieval() ไปสร้างเป็น Object เพื่อนำไปใช้งาน

ตัวอย่างการนำเอา Algorithm ไปใช้งานจริงโดยตัวอย่างนี้จะใช้ภาษา ASP.NET(C#) และระบบฐานข้อมูลเชิงสัมพันธ์ MSSQL แสดงดังต่อไปนี้

```
protected void binddingData(string keyword) {
    string _table; // ชื่อตารางที่ต้องการนำเอา Algorithm ไปใช้งาน
    string[] _displayField = new string[5]; // ชื่อคอลัมท์ที่ใช้แสดงผล
    string[] _ conditionField = new string[1]; // ชื่อคอลัมท์ที่ใช้ทำเงื่อนไข(ตัดข้อมูล)


    _displayField[0] = "research_id";
    _displayField[1] = "research_ref_code";
    _displayField[2] = "research_name_th";
    _displayField[3] = "research_name_eng";
    _displayField[4] = "research_code_nrct_ref";


    _ conditionField[0] = "isActive = 1";


    _table = "data_research ";
    InformationRetrieval ir = new InformationRetrieval(
        _displayField,
        _conditionFillterField,
        _table,
        keyword,
```

```
        "Data Source=ip address:Initial Catalog=db_name;Persist
         Security Info=True:User ID=username:Password=password;");
    try {
        gvSearch.DataSource = ir.DataBind();
        gvSearch.DataBind()
    }
    catch {
         gvSearch.DataSource = null;
        gvSearch.DataBind();
    }
}
```

3.4.2 ส่วนที่ใช้ติดต่อผู้ใช้งาน

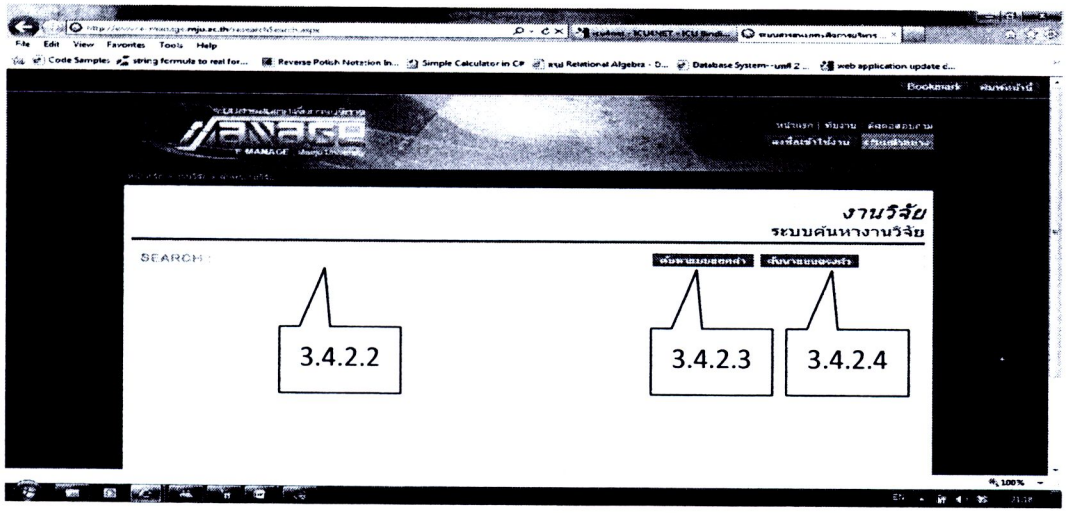
3.4.2.1 แบบฟอร์มค้นหาข้อมูล

3.4.2.2 ช่องป้อนคำหลักในการค้นหา

3.4.2.3 ปุ่มค้นหา

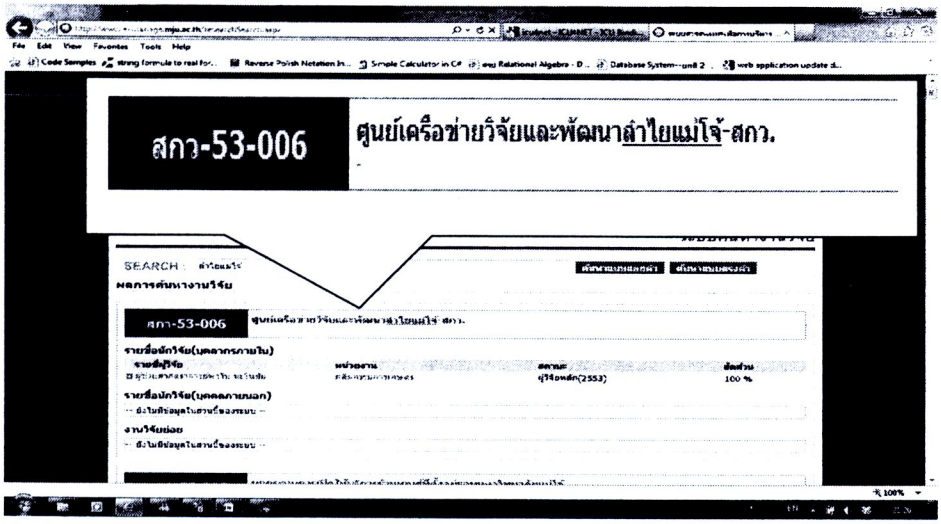
3.4.2.4 ข้อมูลแบบแยกคำ หรือปุ่มที่เลือกใช้งาน Algorithm การค้นคืนสารสนเทศ

3.4.2.5 ปุ่มค้นหาข้อมูลแบบตรงคำ หรือปุ่มที่เลือกใช้งานกระบวนการค้นหาข้อมูลตามมาตรฐานระบบการค้นหาข้อมูลบนระบบฐานข้อมูลเชิงสัมพันธ์




**ภาพ 3.5** แสดงแบบฟอร์มค้นหาข้อมูล

3.4.2.6 ผลลัพธ์ที่ได้จากการค้นหา ตัวอย่างคำหลักที่ใช้ค้นหาคือ “ลำไยแม่โจ้” แสดงดังภาพ 3.6




**ภาพ 3.6** แสดงผลลัพธ์ที่ได้จากการค้นหา