

第十二章 DNA ETL

第一节 DNA ETL 的动机

我在养疗经[17]集成 ETL, 最早准备应用于数据节点计算, 后来在设计功能组件的过程中需要很多配置, 一开始我是做成按钮等组件, 发现占版面面积太多, 于是准备节点化来配置, 我思考中, 如果计算, 处理, 配置都能 ETL 节点化, 养疗经[17]的页面加载, 都用节点编程? 那干脆设计成整体扩展模式加载数据, 于是就这样设计了, 我又思考了下如果每个插件都需要文本文件来进行描述节点应用内容, 干脆语义元基化节点包名, 省去很多事. 想到这, 我有一个动机将 ETL 进行肽化. 于是就是开始了。















DNA ETL 的应用需求

```

v 📁 smallMedicine
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addBuYunBuYuKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addChanHouKangFuKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addChanKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addDanXianKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addErKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addFuKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addNanKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addNanXinJiYinKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addShenNeiKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addWeiChanKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addXinNeiKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addXinShengErKePage
v 📁 neiWaiKe
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addLaoNianKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addMaZuiKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addMiNiaoWaiKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addNaoWaiKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addPuWaiKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addWuGuanKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addXianDaiHuLiPage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addXinWaiKePage
  > 📁 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addXiongWaiKePage
v 📁 PGMedicine
  v 📁 OSI.OSU.SI.OVI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addPGSearchPage
    > 📄 addPGSearchPageNodeASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI.java
    > 📄 addPGSearchPageOSU_MSQ_AOI_AOD_AOU_AOQ_VES.java
  
```

如上图, 目前养疗经[17]的肽化是基于JAVA 文件肽化. 我本想用 C 语言, 但是比较费劲, 就打止了. 底层语言很耗时, ryby, C#, GOLANG, C++, python 肽化还是方便的 汇编和 c 就算了..

第三节 DNA ETL 的具体描述

 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addFuChanBook.jar	2020/12/29 22:33	Executable Jar File	974 KB
 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addHLSBook.jar	2020/12/29 23:59	Executable Jar File	5,355 KB
 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addNankKeRW9Book.jar	2020/12/29 22:34	Executable Jar File	110 KB
 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addXinShengErBook.jar	2020/12/29 22:34	Executable Jar File	974 KB
 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addXYJZBook.jar	2020/12/30 0:01	Executable Jar File	415 KB
 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addXYNKBook.jar	2020/12/30 0:02	Executable Jar File	1,823 KB
 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addXYSCBook.jar	2020/12/29 22:35	Executable Jar File	785 KB
 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addXYWKBook.jar	2020/12/30 0:04	Executable Jar File	1,074 KB
 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addZYBCBook.jar	2021/2/9 18:26	Executable Jar File	1,203 KB
 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addZYNKBook.jar	2020/12/30 0:06	Executable Jar File	2,056 KB
 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addZYWSBook.jar	2020/12/30 0:07	Executable Jar File	207 KB
 OSI.OSU.SI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addZYZDBook.jar	2020/12/30 0:09	Executable Jar File	312 KB
 OSI.OSU.SI.OVI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addPGSearchPage.jar	2021/1/11 5:38	Executable Jar File	51 KB
 OSI.OSU.SI.OVI.OSI.AVI.AEI.ACI.ASI.OVI.OEI.OCI.OSI.PVI.PEI.PCI.PSI.addZNSZPage.jar	2020/12/29 22:45	Executable Jar File	108 KB

这种肽插件结构. 目前养疗经[17]导入没有出现异常, 比较稳定实用, 我于是开始思考通过肽展计算在标识中加关键字, 隐写数据, 因为隐写术属于特工情报学, 涉及国家安全问题, 第二卷不介绍.

下面的插件节点代码通过语料库表, 将文件进行了元基肽化, 如果想翻译, 可以通过元基TVM 解码变回来即可, 这个语料库和语法思想, 不属于开源和全民统一使用, 第二卷不做完整描述. 语料库中有片段介绍.

```

package OSI. OSU. SI. OVI. SQ. AVQ. SI. SD. SU. SQ. ASU. OSU. PSU. MSU. AVQ. ASQ. dNA3DShow;

import java. awt. *;

import java. io. FileNotFoundException;

import java. io. IOException;

import java. util. Map;

import javax. swing. *;

import org. ASQ. PSU. OCI. tinos. engine. analysis. Analyzer;

import ME. sample. App;

import OSI. OSU. AOPM. VECS. IDUQ. GUI. Flash. ThisCanvas;

import OSI. OSU. OEQ. MCQ. GUI. OSGI. OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI;

public class dNA3DShowNodeASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI extends
OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI{

//   JTextPane text;

//   Object[][] tableData_old;

//   public App u;

//   public Analyzer analyzer;

//   public Map<String, String> pos;

public dNA3DShowNodeASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI() throws
    IOException{ thisIcon= new ImageIcon(this. getClass(). getResource("dNA3DShow. jpg"));

    SQ_OSU_MSQ_OSU_AVQ_ASQ_SQ_VPC_PCS= new String("DNA 染色体 三维显示");

    AMV_MVS_VSQ= new String("HUMANOID");

    Image img= ((ImageIcon) thisIcon). getImage();

    Image newimg= img. getScaledInstance(30, 30, java. awt. Image. SCALE_SMOOTH);

    thisImage= img. getScaledInstance(30, 30, java. awt. Image. SCALE_SMOOTH );

    thisIcon= new ImageIcon(newimg);

//

```

```

    }

    public dNA3DShowNodeASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI(Object[][] tableData_old, JTextPane text, App
u
        , Analyzer analyzer, Map<String, String> pos) throws
IOException{ this. text= text;
    this. tableData_old= tableData_old;
    thisIcon= new ImageIcon(this. getClass(). getResource("dNA3DShow. jpg"));
    SQ_OSU_MSQ_OSU_AVQ_ASQ_SQ_VPC_PCS= new String("DNA 染色体 三维显示");
    AMV_MVS_VSQ= new String("HUMANOID");
    Image img= ((ImageIcon) thisIcon). getImage();
    Image newimg= img. getScaledInstance(30, 30, java. awt. Image. SCALE_SMOOTH);
    thisImage= img. getScaledInstance(30, 30, java. awt. Image. SCALE_SMOOTH );
    thisIcon= new ImageIcon(newimg);
    //
    this. u= u;
    this. analyzer= analyzer;
    this. pos= pos;
}

    public void MEI_MSU(JTextPane jTextPane, ThisCanvas canvas) throws IOException{
        SQ_OSU_MSQ_OSU_AVQ_ASQ_AVQ_ASQ_OVQ_OSQ_VSQ = new
dNA3DShowOSU_MSQ_AVQ_ASQ_OVQ_OSQ_VSQ();
        SQ_OSU_MSQ_OSU_AVQ_ASQ_OPE_OPC_ECI = new dNA3DShowOSU_MSQ_OPE_OPC_ECI();
        SQ_OSU_MSQ_OSU_AVQ_ASQ_AOI_AOD_AOU_AOQ_VES = new
dNA3DShowOSU_MSQ_AOI_AOD_AOU_AOQ_VES((dNA3DShowOSU_MSQ_OPE_OPC_ECI)
SQ_OSU_MSQ_OSU_AVQ_ASQ_OPE_OPC_ECI, this. text, this. tableData_old);
        SQ_OSU_MSQ_OSU_AVQ_ASQ_AOI_AOD_AOU_AOQ_VES. config();
        showed = false;
    }

    public void OPE_E(JTextPane jTextPane) throws FileNotFoundException, IOException{

```

```

        ((dNA3DShowOSU_MSQ_OPE_OPC_ECI) SQ_OSU_MSQ_OSU_AVQ_ASQ_OPE_OPC_ECI).
run((dNA3DShowOSU_MSQ_AVQ_ASQ_OVQ_OSQ_VSQ)
SQ_OSU_MSQ_OSU_AVQ_ASQ_AVQ_ASQ_OVQ_OSQ_VSQ);
    }

    public void AVQ_ASQ_OVQ_OSQ_VSQ(JTextPane jTextPane) throws
        Exception{ ((dNA3DShowOSU_MSQ_AVQ_ASQ_OVQ_OSQ_VSQ)
SQ_OSU_MSQ_OSU_AVQ_ASQ_AVQ_ASQ_OVQ_OSQ_VSQ). analyzer= analyzer;
        ((dNA3DShowOSU_MSQ_AVQ_ASQ_OVQ_OSQ_VSQ)
SQ_OSU_MSQ_OSU_AVQ_ASQ_AVQ_ASQ_OVQ_OSQ_VSQ). u= u;
        ((dNA3DShowOSU_MSQ_AVQ_ASQ_OVQ_OSQ_VSQ)
SQ_OSU_MSQ_OSU_AVQ_ASQ_AVQ_ASQ_OVQ_OSQ_VSQ). pos= pos;
        ((dNA3DShowOSU_MSQ_AVQ_ASQ_OVQ_OSQ_VSQ)
SQ_OSU_MSQ_OSU_AVQ_ASQ_AVQ_ASQ_OVQ_OSQ_VSQ). DNNtext=
((dNA3DShowOSU_MSQ_AOI_AOD_AOU_AOQ_VES)
SQ_OSU_MSQ_OSU_AVQ_ASQ_AOI_AOD_AOU_AOQ_VES). DNNtext;
        SQ_OSU_MSQ_OSU_AVQ_ASQ_AVQ_ASQ_OVQ_OSQ_VSQ. view();
        showed = true;
    }

    public OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI AOP_MVE_CSI_DUQ() throws
CloneNotSupportedException, IOException{
        SQ_OSU_MSQ_OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI = new
dNA3DShowNodeASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI(this. tableData_old, this. text, this. u, this. analyzer, this.
pos);

        return SQ_OSU_MSQ_OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI;
    }
}

```

上面这段函数与下面的函数进行验证导入, 可以正确的读取肽插件. 我设计它一开始是为了做肽化版本的 OSGI, 现在我的动机是让肽插件有自主思维意识, 以后通过肽展公式能自主识别插件来补充, 我定义为养疗经[17]的学习功能.

第四节 DNA ETL 的应用实现

```
> medReport
> Listen
> suggest
v NeroTheme
  v OSI.OSU.AOPM.VECS.IDUQ.GUI.Flash
    v Flash.java
      v GUISample.java
        v GUISample
          v ThisCanvas.java
        v OSI.OSU.LYG.vpcs.hallKeeper
        v OSI.OSU.LYG.vpcs.sets
        v OSI.OSU.LYG.vpcs.skivvy
        v OSI.OSU.LYG.vpcs.vision
        v OSI.OSU.MSI.OEI.document.save
        v OSI.OSU.MSQ.GUI.nodeInfo
        v OSI.OSU.MSQ.GUI.nodeProject
        v OSI.OSU.MSQ.sets.stable
        v OSI.OSU.MVU.OVU.GUI.nodeEdit.controller
        v OSI.OSU.OED.PVD.document.delete
        v OSI.OSU.OEI.PVI.document.load
        v OSI.OSU.OEQ.MCQ.GUI.OSGI
        v OSI.OSU.OSQ.MVQ.GUI.theme.neroCell
        v OSI.OSU.OVQ.MSQ.GUI.platForm
        v OSI.OSU.OVU.MVQ.GUI.nodeView
        v OSI.OSU.OVU.MVU.GUI.nodeEdit
        v OSI.OSU.PSQ.OEU.document.neroCell
        v OSI.OSU.SI.MCI.OEI.GUI.extOSGI

//我准备之后设计成病毒式热插拔, 因为绕过虚拟机的思想涉及情报学特工和计算机病毒领域
//, 害怕国家相关安全体系管控, 暂时不研发。
OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI
= (OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI) myobject;
if(nodeReflection.containsKey(OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI.SQ_OSU_MSQ_OSU_AVQ_ASQ_SQ_VPC_PCS)) {
    rightBotJTextPane.setText("请勿重复添加, 节点已存在"+ System.currentTimeMillis());
    continue Here;
    //return;
}
nodeReflection.put(OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI.SQ_OSU_MSQ_OSU_AVQ_ASQ_SQ_VPC_PCS, null);
OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI.register(tableData_old, text, u, analyzer, pos);
try {
    //search register need add the information
    if(OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI.SQ_OSU_MSQ_OSU_AVQ_ASQ_SQ_VPC_PCS.contains("添加")
        && OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI.SQ_OSU_MSQ_OSU_AVQ_ASQ_SQ_VPC_PCS.contains("页")) {
        u.searchList.add(OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI); //memory increment
    }
    //searchList
    //
    OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI.IMP_PSU();
    rightBotJTextPane.setText("节点:"
        + OSU_AVQ_ASQ_ASQ_OCQ_OSI_PCI_PCU_MCI_MCU_MSI.SQ_OSU_MSQ_OSU_AVQ_ASQ_SQ_VPC_PCS
        + "已经更新成功"+ System.currentTimeMillis()
        + "\r\n"+ rightBotJTextPane.getText());
    rightBotJTextPane.validate();
    thread.sleep(100);
} catch (IOException e1) {
    // TODO Auto-generated catch block
    e1.printStackTrace();
} catch (InterruptedException e2) {
    // TODO Auto-generated catch block
```