智能多轮对话机器人开发实战

主讲：张子良

系统性：系统方法、结构严谨，知识点全面覆盖；

完整性：完整案例、场景驱动，开发过程步步到位；

实战性：实战指引、手脑齐动，工程案例实战实操；

源码性：源码实训，开源框架、完整案例源码程序；

第四部分

第四部分 任务式智能对话机器人

- 任务式智能对话机器人基础
- 任务式智能对话机器人案例
- 任务式智能对话机器人实战
- 任务式智能对话机器人源码

1.1 典型案例-业务场景：网络订票

1.特定领域

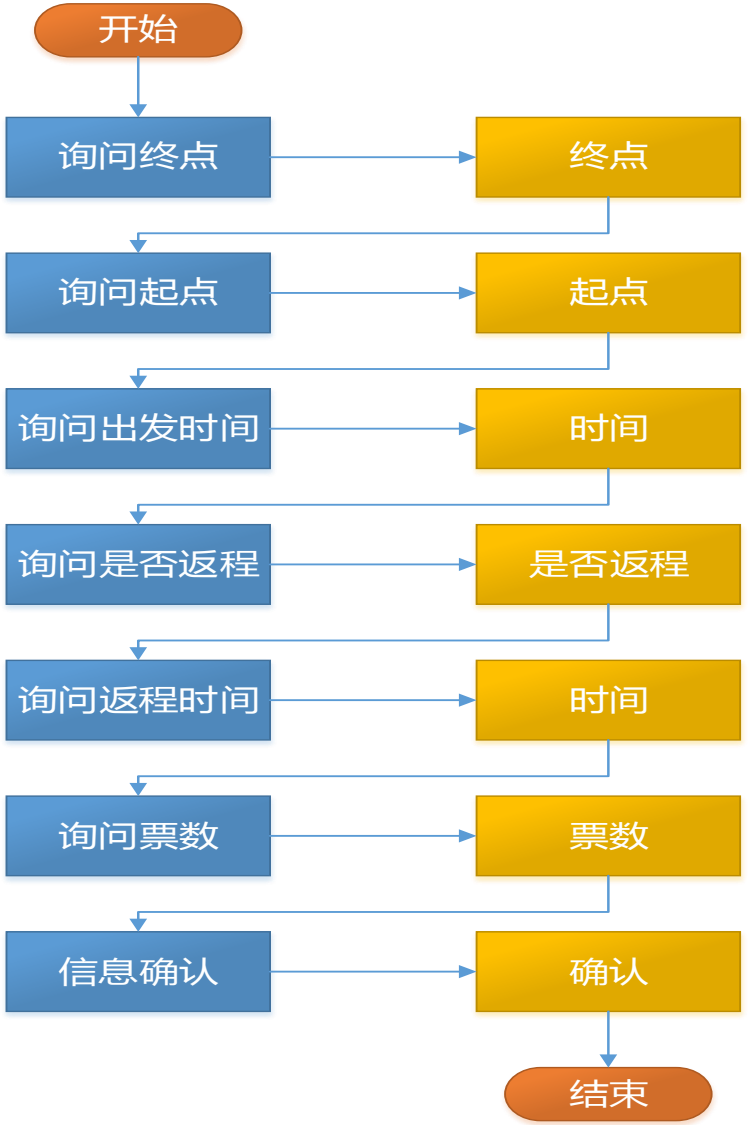
2.目标明确

3.多轮交互

4.流程稳定

1.1 典型案例-业务流程

机器人



人

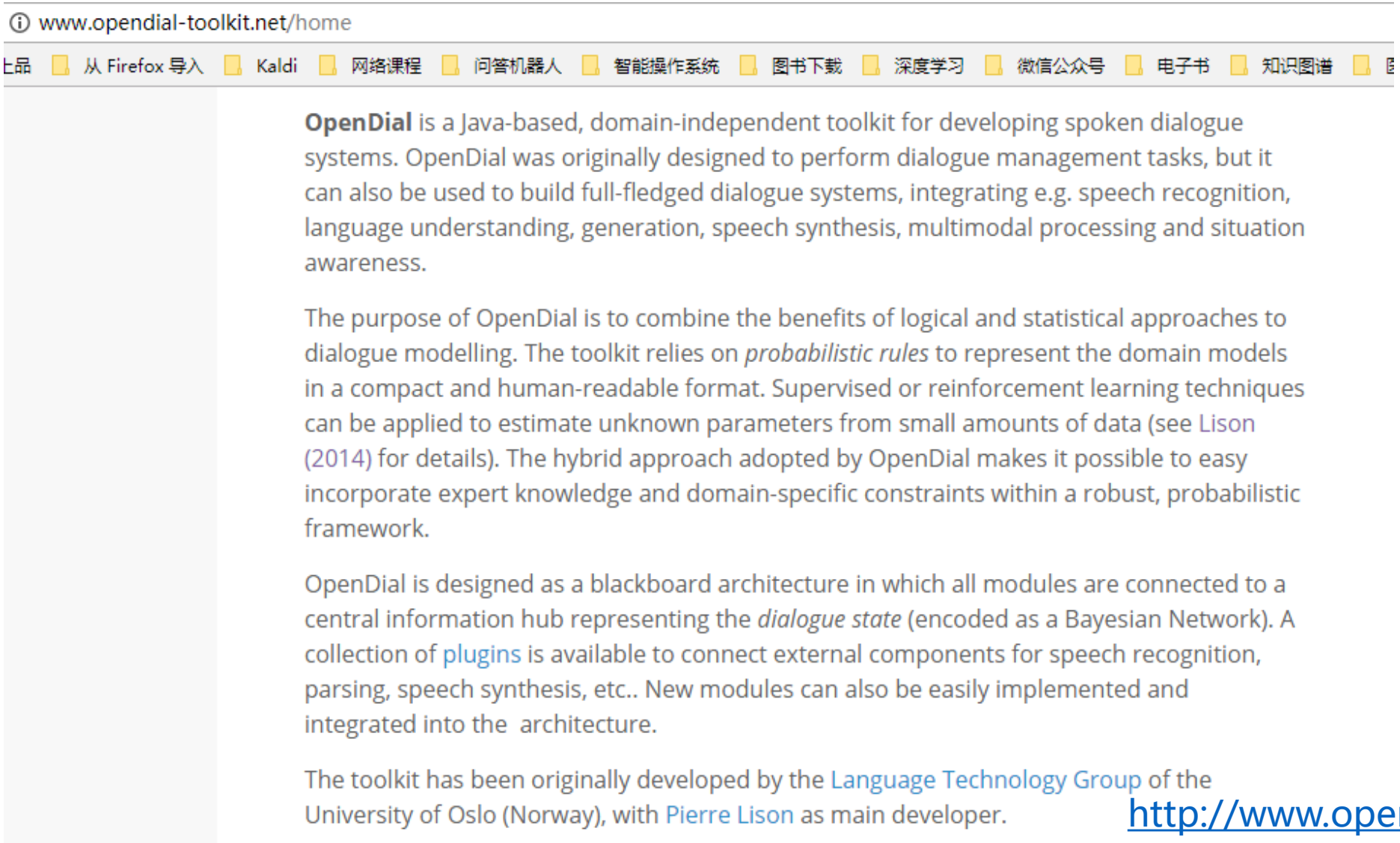
1.1 典型案例-效果演示

[Jackson]	"欢迎访问智能航空订票系统,我是您的私人票务管家。"
[Jackson]	请问您要去那个城市？
[Your]	北京
[Jackson]	好的，目的地：北京.
[Jackson]	请问您打算从那个城市出发？
[Your]	上海
[Jackson]	好的,出发自： 上海.
[Jackson]	请问，您打算什么时候，前往目的地:北京？
[Your]	3月2日
[Jackson]	好的,于3月 2，从上海起飞，去往 北京.
[Jackson]	需要帮您预订返程的机票吗？
[Your]	是的
[Jackson]	您计划什么时候返回:上海?
[Your]	3月10日
[Jackson]	好的，于 3月 10自 北京 飞往 上海 .
[Jackson]	于3月 2从上海出发至北京，返程日期为：3月 10，当日票价为：299 人民币/张.需要我帮您预订吗?
[Your]	好的
[Jackson]	抱歉，您可以重复一次吗?
[Your]	是
[Jackson]	请问您需要预定几张机票？
[Your]	2

1.1 典型案例-效果演示

[Jackson]	请问您需要预定几张机票？
[Your]	2
[Jackson]	好的, 2 张机票.
[Jackson]	您已经预定 2 往返机票，自 上海 至 北京 于3月 2 并且 返程于 3月 10 总价： 598 元人民币.确认下单吗?
[Your]	Yes
[Jackson]	好的,您的机票已经完成预定,将会发送到您的邮箱.
[Jackson]	请问还有什么可以帮您的吗？
[Your]	No
[Jackson]	非常感谢您的支持，再见。

1.2 开发环境安装部署：opendial官网（ http://www.opendial-toolkit.net/ ）



http://www.opendial-toolkit.net/

1.2 开发环境安装部署：opendial下载

1. Downloading OpenDial

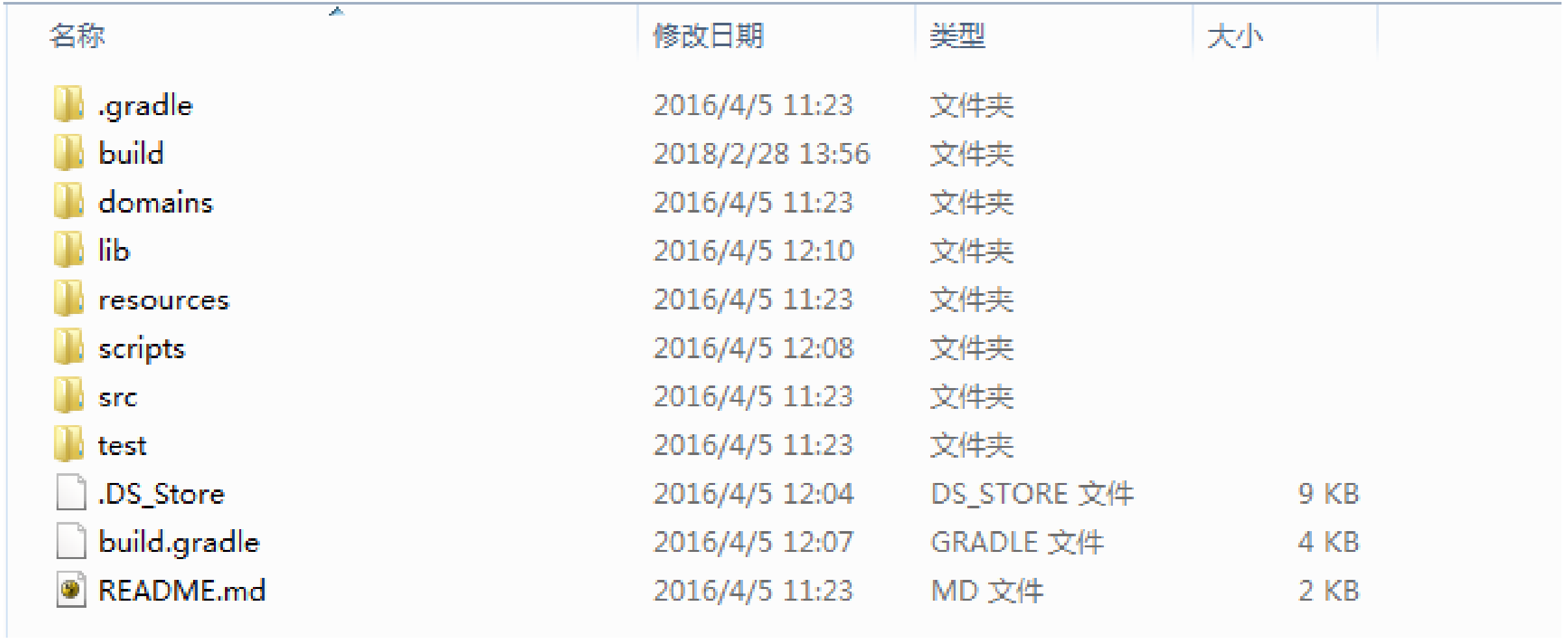
The Download page contains the various OpenDial releases.

System requirements:

- **Java 8**
- Gradle (if you want to modify or extend the source code)

NB: If you are using Maven/Gradle for your software and are only interested in fetching OpenDial as an external dependency, there a Maven package for Opendial on Jcenter (`groupId: opendial,artifactId: opendial`).

1.2 开发环境安装部署：opendial解压

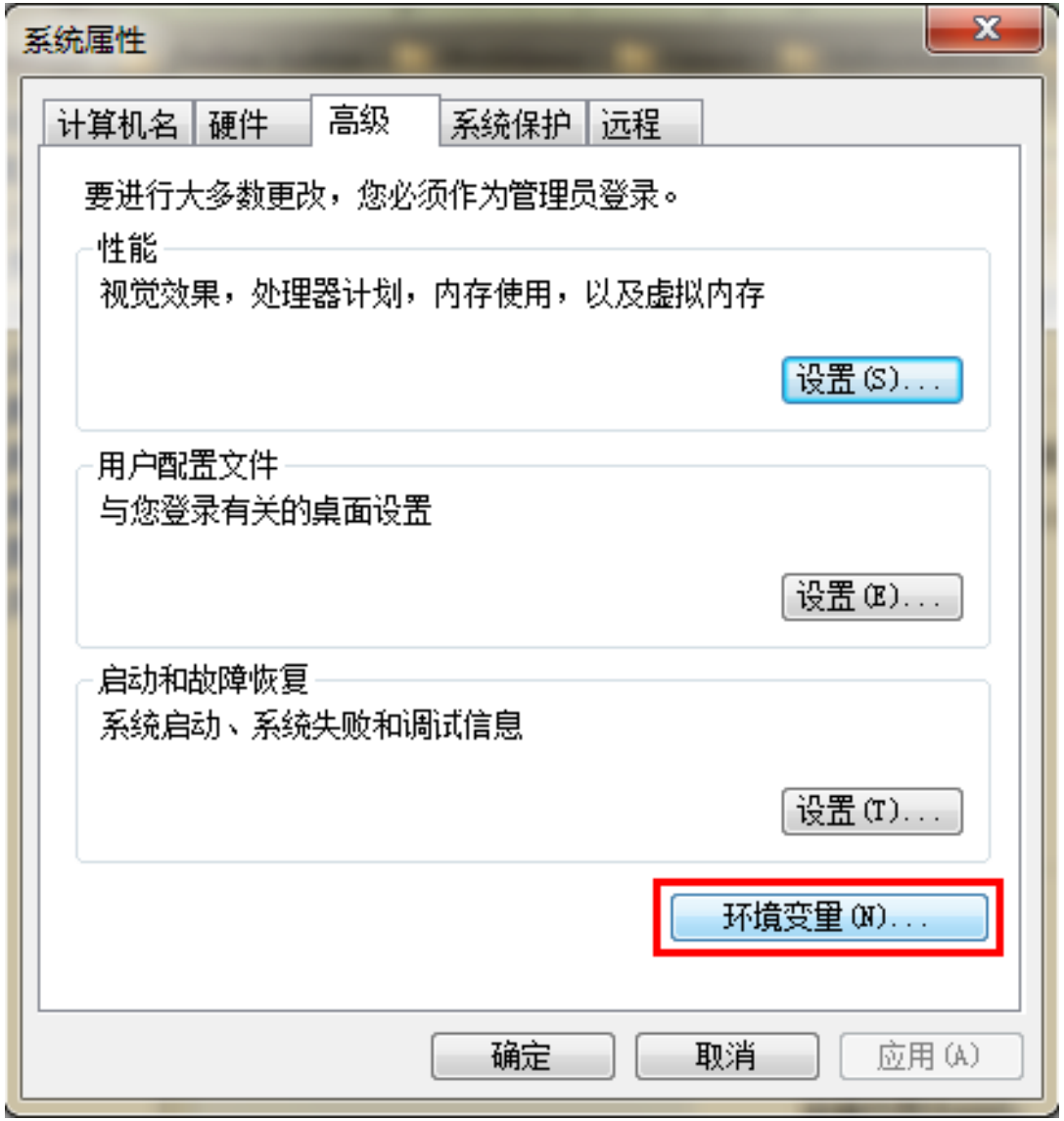


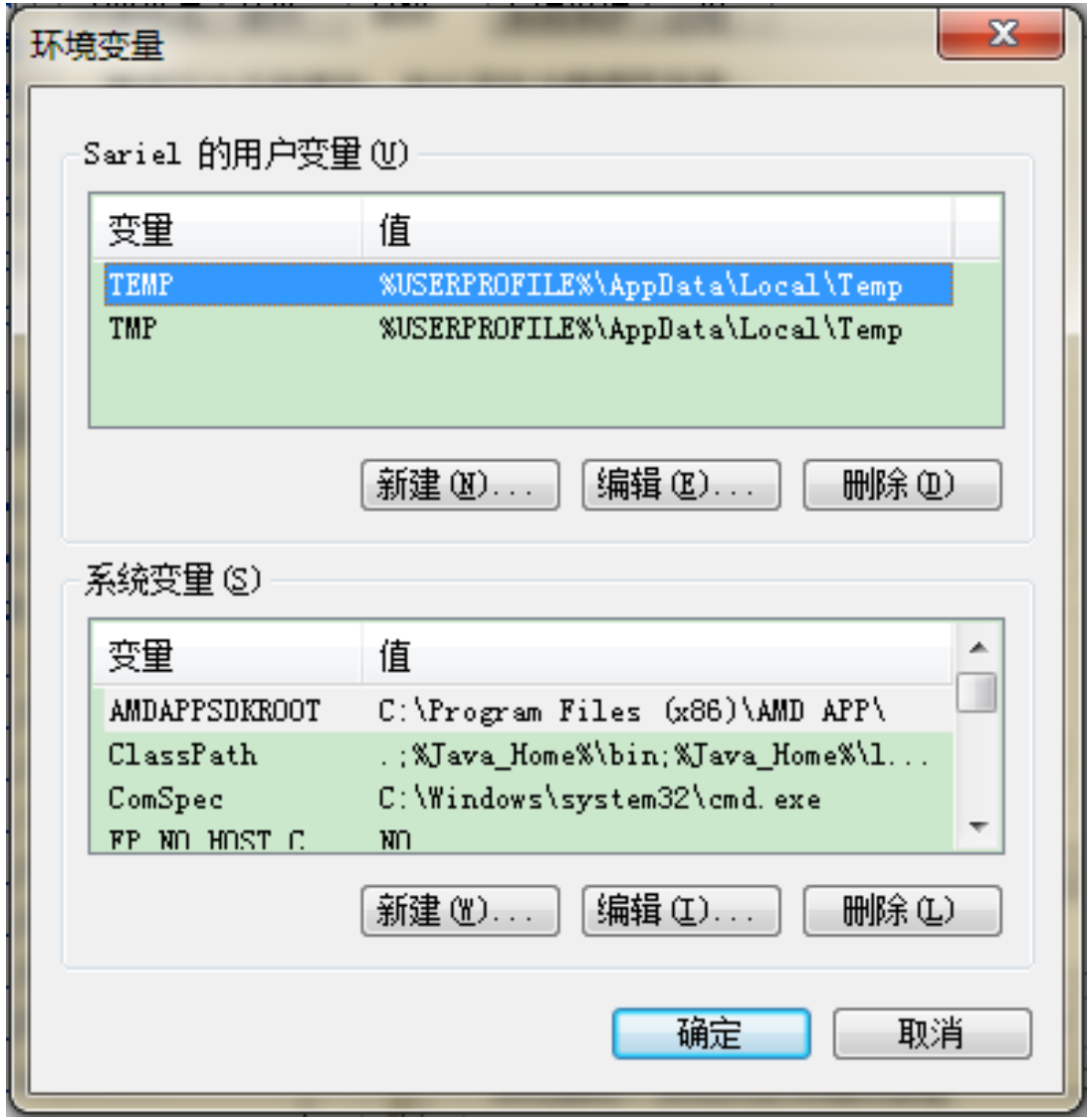
名称	修改日期	类型	大小
.gradle	2016/4/5 11:23	文件夹	
build	2018/2/28 13:56	文件夹	
domains	2016/4/5 11:23	文件夹	
lib	2016/4/5 12:10	文件夹	
resources	2016/4/5 11:23	文件夹	
scripts	2016/4/5 12:08	文件夹	
src	2016/4/5 11:23	文件夹	
test	2016/4/5 11:23	文件夹	
.DS_Store	2016/4/5 12:04	DS_STORE 文件	9 KB
build.gradle	2016/4/5 12:07	GRADLE 文件	4 KB
README.md	2016/4/5 11:23	MD 文件	2 KB

1.2 开发环境安装部署：opendial安装-Java8

1.2 开发环境安装部署：设置环境变量

1.2 开发环境安装部署：设置环境变量





1.2 开发环境安装部署：设置环境变量

在"系统变量"中设置3项属性，JAVA_HOME,PATH,CLASSPATH(大小写无所谓),若已存在则点击"编辑"，不存在则点击"新建"。

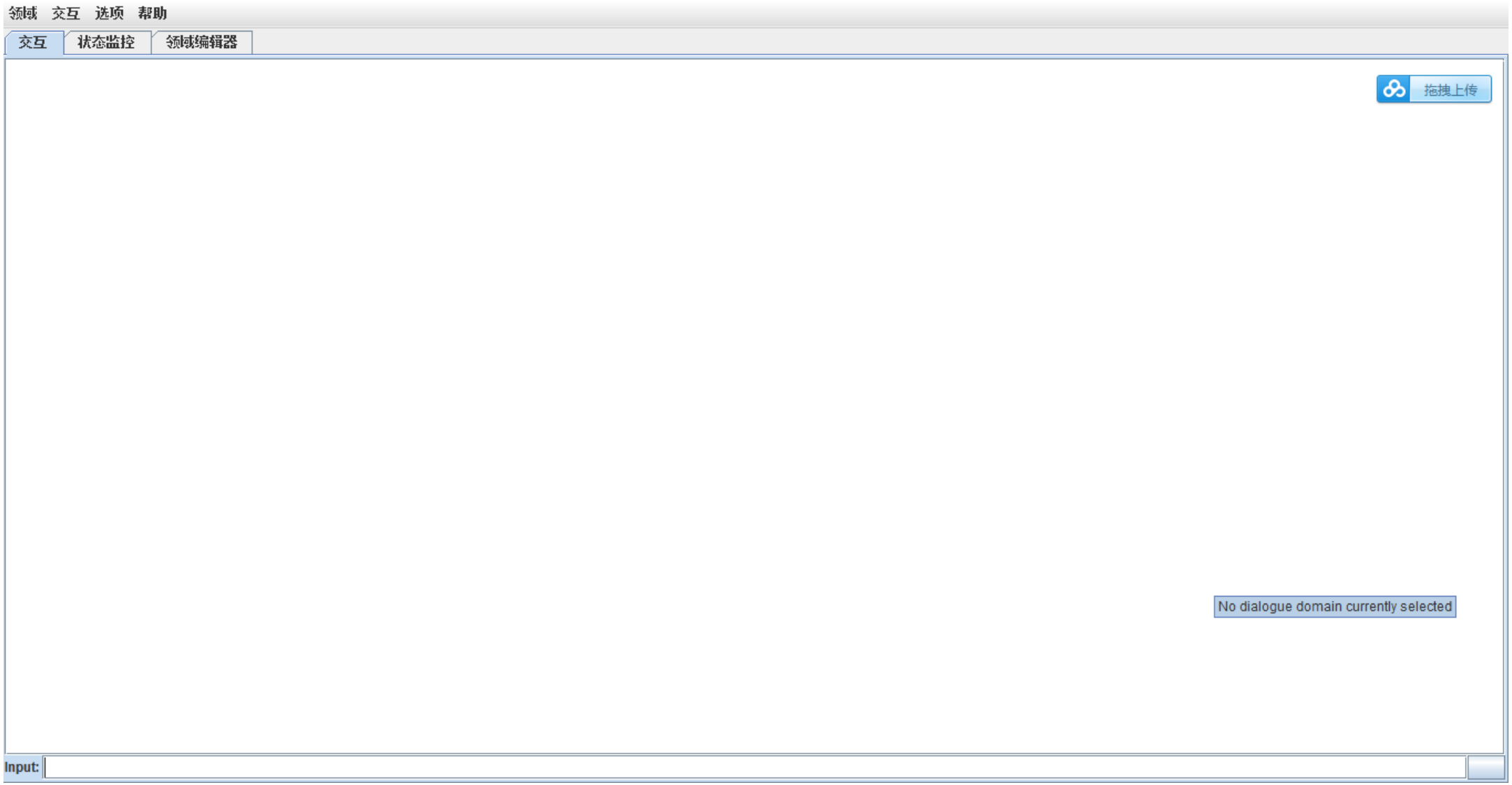
变量设置参数如下：

- 变量名：JAVA_HOME
- 变量值：C:\Program Files (x86)\Java\jdk1.8.0_91　　// 要根据自己的实际路径配置
- 变量名：CLASSPATH
- 变量值：.;%JAVA_HOME%\lib\dt.jar;%JAVA_HOME%\lib\tools.jar;　　//记得前面有个"."
- 变量名：Path
- 变量值：%JAVA_HOME%\bin;%JAVA_HOME%\jre\bin;

注意：在 Windows10 中，因为系统的限制，path 变量只可以使用 JDK 的绝对路径。%JAVA_HOME% 会无法识别，导致配置失败。如下所示：

```
C:\Program Files (x86)\Java\jdk1.8.0_91\bin;C:\Program Files (x86)\Java\jdk1.8.0_91\jre\bin;
```

1.2 开发环境安装部署：运行



第四部分

第四部分 任务式智能对话机器人

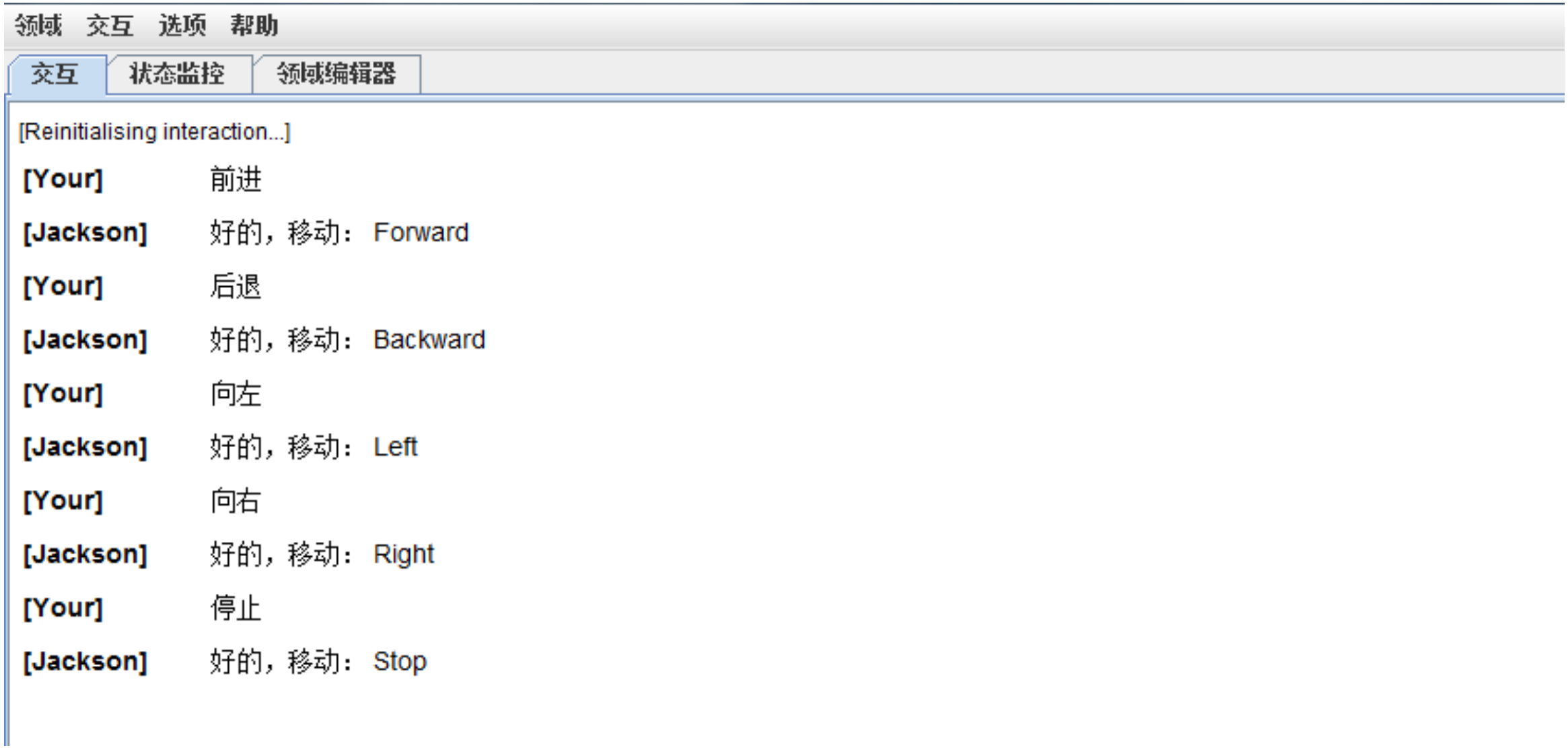
- 任务式智能对话机器人基础
- 任务式智能对话机器人案例
- 任务式智能对话机器人实战
- 任务式智能对话机器人源码

2.1 案例1：声控机器人-业务场景

机器人控制指令：

- 前进
- 后退
- 向左
- 向右
- 停止

2.1 案例1：声控机器人-效果演示



2.1 案例1：声控机器人-配置文件（用户理解模型）

```
<!-- NLU model -->
<model trigger="u_u">
    <rule>
    <case>
      <condition operator="or">
        <if var="u_u" value="*左转*" relation="contains" />
        <if var="u_u" value="*向左*" relation="contains" />
        <if var="u_u" value="*往左*" relation="contains" />
      </condition>
      <effect prob="1">
        <set var="a_u" value="Request(Left)" />
      </effect>
    </case>
    <case>
      <condition operator="or">
        <if var="u_u" value="*右转*" relation="contains" />
        <if var="u_u" value="*向右*" relation="contains" />
        <if var="u_u" value="*往右*" relation="contains" />
      </condition>
      <effect prob="1">
        <set var="a_u" value="Request(Right)" />
      </effect>
    </case>
    <case>
      <condition operator="or">
        <if var="u_u" value="*向前*" relation="contains" />
        <if var="u_u" value="*直行*" relation="contains" />
        <if var="u_u" value="*前进*" relation="contains" />
      </condition>
      <effect prob="1">
        <set var="a_u" value="Request(Forward)" />
      </effect>
    </case>
```

2.1 案例1：声控机器人-配置文件（用户理解模型）

```
    <case>
      <condition operator="or">
        <if var="u_u" value="*向后*" relation="contains" />
        <if var="u_u" value="*倒退*" relation="contains" />
        <if var="u_u" value="*后退*" relation="contains" />
      </condition>
      <effect prob="1">
        <set var="a_u" value="Request(Backward)" />
      </effect>
    </case>
    <case>
      <condition>
        <if var="u_u" value="停止" relation="contains" />
        <if var="u_u" value="暂停" relation="contains" />
        <if var="u_u" value="stop" relation="contains" />
      </condition>
      <effect prob="1">
        <set var="a_u" value="Request(Stop)" />
      </effect>
    </case>
  </rule>

</model>
```

2.1 案例1：声控机器人-配置文件（用户行为模型）

```
<!-- 行为选择模型 -->
<model trigger="a_u">
  <rule id="movement">
    <case>
      <condition>
        <if var="a_u" value="Request({X})" />
      </condition>
      <effect util="1">
        <set var="a_m" value="Move({X})" />
      </effect>
    </case>
  </rule>

  <rule id ="negative">
    <case>
      <effect util="-0.5">
        <set var="a_m" value="Move(*)" />
      </effect>
    </case>
  </rule>
  <rule id="repeat">
    <case>
      <effect util="0.2">
        <set var="a_m" value="AskRepeat" />
      </effect>
    </case>
  </rule>

</model>
```

2.1 案例1：声控机器人-配置文件（机器行为模型）

```
<!-- 行为生成模型 -->
<model trigger="a_m">

  <rule>
    <case>
      <condition>
        <if var="a_m" value="Move({X})" />
      </condition>
      <effect util="1">
        <set var="u_m" value="好的，移动：  {X}" />
      </effect>
    </case>
    <case>
      <condition>
        <if var="a_m" value="AskRepeat" />
      </condition>
      <effect util="1">
        <set var="u_m" value="抱歉，我没有听清。请重复一遍好吗？" />
      </effect>
    </case>
  </rule>
</model>
```

2.2 案例2：智能闹钟-业务场景

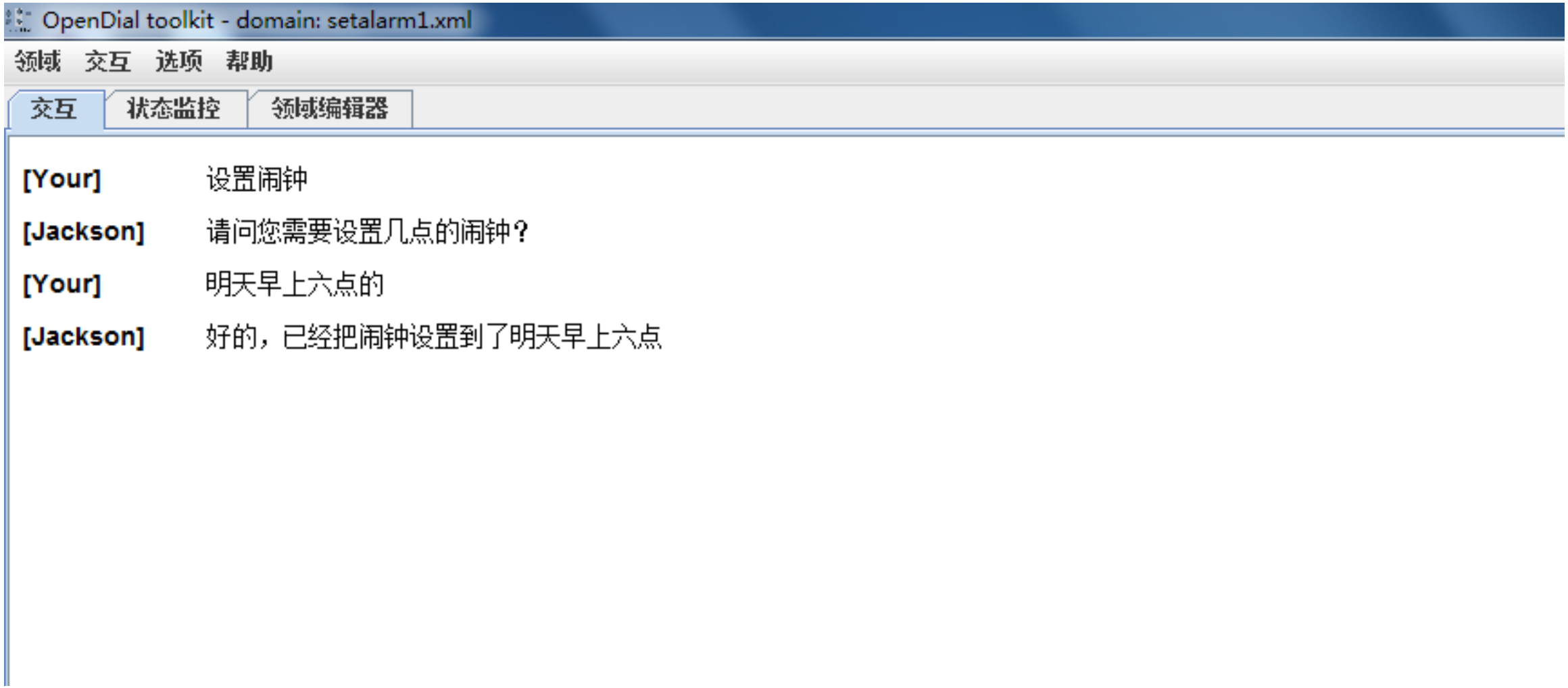
用户：设置闹钟

Siri：请问您需要设置几点的闹钟？

用户：明天早上六点的

Siri：好的，已经把闹钟设置到了明天早上六点

2.2 案例2：智能闹钟-演示效果



2.2 案例2：智能闹钟-配置文件（用户理解模型）

```
<?xml version="1.0" encoding="gb2312"?>
 <domain>
  <model trigger="u_u">
    <rule>
      <case>
        <condition>
          <if var="u_u" relation="=" value="设置闹钟"/>
        </condition>
        <effect prob="1">
          <set var="a_u" value="SetAlarm"/>
        </effect>
      </case>
      <case>
        <condition>
          <if var="u_u" relation="=" value="明天早上六点的"/>
        </condition>
        <effect prob="1">
          <set var="a_u" value="InformTime"/>
        </effect>
      </case>
    </rule>
  </model>
```

2.2 案例2：智能闹钟-配置文件（用户行为模型）

```
<model trigger="a_u">
  <rule>
    <case>
      <condition>
        <if var="a_u" relation="=" value="SetAlarm"/>
      </condition>
      <effect prob="1">
        <set var="a_m" value="RequestTime"/>
      </effect>
    </case>
    <case>
      <condition>
        <if var="a_u" relation="=" value="InformTime"/>
      </condition>
      <effect prob="1">
        <set var="a_m" value="ToDoSetAlarm"/>
      </effect>
    </case>
  </rule>
</model>
```

2.2 案例2：智能闹钟-配置文件（机器行为模型）

```
<model trigger="a_m">
  <rule>
    <case>
      <condition>
        <if var="a_m" relation="=" value="RequestTime"/>
      </condition>
      <effect prob="1">
        <set var="u_m" value="请问您需要设置几点的闹钟？ "/>
      </effect>
    </case>
    <case>
      <condition>
        <if var="a_m" relation="=" value="ToDoSetAlarm"/>
      </condition>
      <effect prob="1">
        <set var="u_m" value="好的，已经把闹钟设置到了明天早上六点.">
      </effect>
    </case>
  </rule>
</model>
```

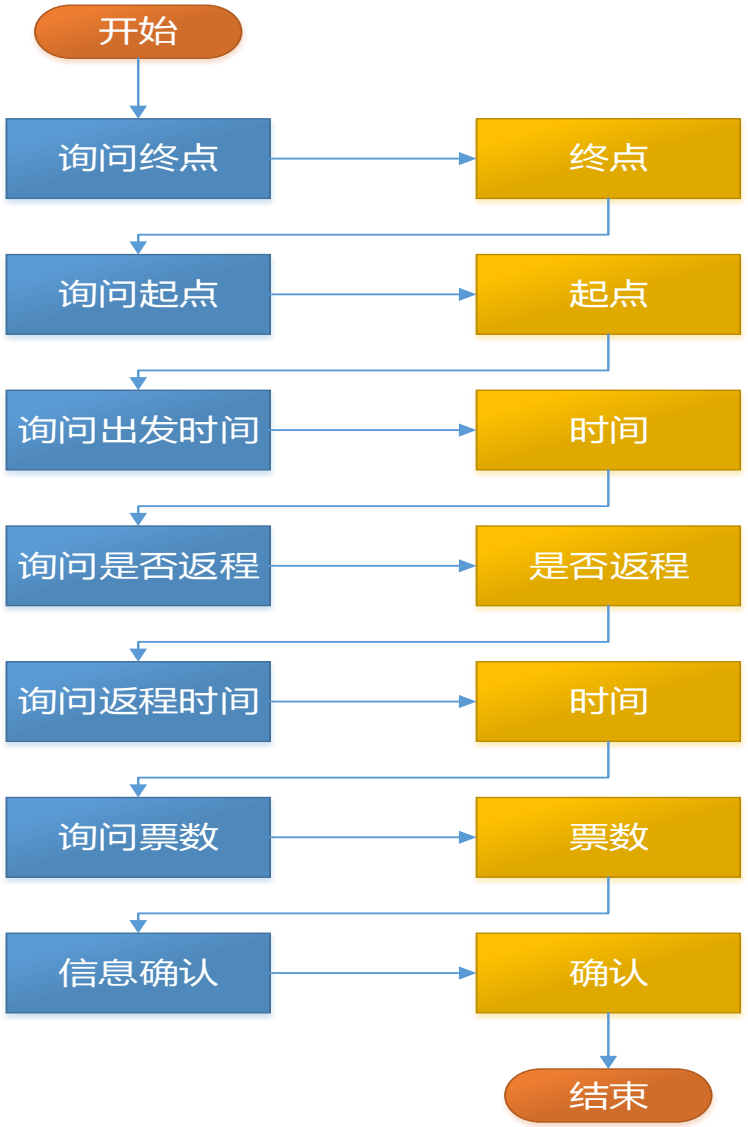
第四部分

第四部分 任务式智能对话机器人

- 任务式智能对话机器人基础
- 任务式智能对话机器人案例
- 任务式智能对话机器人实战
- 任务式智能对话机器人源码

3.1 实战案例：航空订票机器人语料库（引导文件）

机器人



人

3.1 实战案例：航空订票机器人语料库（引导文件）

3.1 实战案例：航空订票机器人语料库（引导文件）

```
<domain>
  <initialstate>

    <!-- Starting prompt -->
    <variable id="u_m" >
      <value>"欢迎访问智能航空订票系统,我是您的私人票务管家：Jackson."</value>
    </variable>

    <!-- We start the dialogue by asking for the destination -->
    <variable id="current_step">
      <value>Destination</value>
    </variable>

  </initialstate>

  <!-- Natural language understanding models -->
  <import href="example-flightbooking_nlu_cn.xml"/>

  <!--  Dialogue management models (action selection and transition) -->
  <import href="example-flightbooking_dm_cn.xml"/>

  <!-- Natural language generation models -->
  <import href="example-flightbooking_nlg_cn.xml"/>


  <!-- External module showing how to interface the dialogue system with an external
  database (used here to find the ticket prices and "book" the tickets) -->
  <settings>
    <modules>opendial.modules.examples.FlightBookingExample</modules>
  </settings>
</domain>
```

3.2 实战案例：航空订票机器人语料库（NLU-用户行为识别）

```
<model trigger="u_u">
  <!-- This model takes the raw user utterance as input, and outputs the corresponding
  dialogue acts from the user. This model uses simple shallow patterns to extract
  domain-specific information (such as airports or dates) from the utterance. -->

  <!-- Extracts booking information related to the departure or destination  -->
  <rule>
    <case>
      <condition>
        <if var="Airport" relation="in"
          value="[北京,上海,天津,石家庄,济南,哈尔滨,重庆,
              长春,沈阳,呼和浩特,乌鲁木齐,兰州,西宁,西安,银川,郑州,太原,合肥,长沙,昆明,武汉,南京,成都,贵阳,昆明,
              拉萨,杭州,南昌,广州,福州,台北,海口,香港,澳门,南宁]" />
        <if var="u_u" relation="contains" value="(to|from)? {Airport}" />
      </condition>
      <effect>
        <set var="a_u" value="Inform(Airport,{Airport})" exclusive="false"/>
      </effect>
    </case>
  </rule>

  <!-- Extracts booking information related to the flight dates.  -->
  <rule>
    <case>
      <condition>
        <if var="Month" relation="in"
          value="[1月,2月,3月,4月,5月,6月,7月,8月,9月,10月,11月,12月]" />
        <if var="u_u" relation="contains" value="(on)? {Month}{Day}日" />
        <if var="Day" relation="&gt;" value="0" />
        <if var="Day" relation="&lt;" value="32" />
      </condition>
      <effect>
        <set var="a_u" value="Inform(Date,{Month},{Day})" exclusive="false"/>
```

3.2 实战案例：航空订票机器人语料库（NLU-槽位填充）

```
<model trigger="a_u">
  <!-- 用户行为This model take the dialogue act fom the user and uses it to fill the corresponding
  slots (departure, destination, flight dates, number of tickets, etc.). -->
  <!-- Fills the slots in accordance with the information in the dialogue act -->
  <rule>
    <case>
      <condition>
        <if var="current_step" value="(Destination|Departure)" />
        <if var="a_u" relation="contains" value="Inform(Airport,{Airport})" />
      </condition>
      <effect>
        <set var="{current_step}" value="{Airport}" />
      </effect>
    </case>
    <case>
      <condition>
        <if var="current_step" value="(Date|ReturnDate)" />
        <if var="a_u" relation="contains" value="Inform(Date,{Month},{Day})" />
      </condition>
      <effect>
        <set var="{current_step}" value="{Month} {Day}" />
      </effect>
    </case>
    <case>
      <condition>
        <if var="current_step" value="NbTickets" />
        <if var="a_u" relation="contains" value="Inform(Number,{Number})" />
      </condition>
      <effect>
        <set var="NbTickets" value="{Number}" />
      </effect>
    </case>
  </rule>
```

3.3 实战案例：航空订票机器人语料库（DM-行为选择模型）

```
<model trigger="Destination,Departure,Date,ReturnDate,NbTickets">
  <!-- This model specifies the utilities of various system actions, such
    as clarification requests (repetitions and confirmations) and
    grounding actions. -->

  <!-- If the current step is to ask for the destination, specifies the utilities
    of a confirmation request or a grounding action using the current value of
    the "Destination" slot -->
  <rule>
    <case>
      <condition>
        <if var="current_step" value="Destination" />
      </condition>
      <effect util="5">
        <set var="a_m" value="Ground(Destination,{Destination})" />
      </effect>
      <effect util="0.5">
        <set var="a_m" value="Confirm(Destination,{Destination})" />
      </effect>
    </case>
  </rule>

  <!-- If the current step is to ask for the departure, specifies the utilities
    of a confirmation request or a grounding action using the current value of
    the "Departure" slot -->
  <rule>
    <case>
      <condition>
        <if var="current_step" value="Departure" />
      </condition>
      <effect util="5">
        <set var="a_m" value="Ground(Departure,{Departure})" />
```

3.3 实战案例：航空订票机器人语料库（DM-行为选择模型）

```
<model trigger="a_m">
  <!-- Transition model that specifies how the selection of a particular
    system action affects the current dialogue state, in particular the current
    step in the dialogue. -->

  <!-- Moves to the next step if a particular slot has been confirmed -->
  <rule>
    <case>
      <condition>
        <if var="a_m" value="Ground(Destination,*)" />
      </condition>
      <effect prob="1">
        <set var="current_step" value="Departure" />
      </effect>
    </case>
    <case>
      <condition>
        <if var="a_m" value="Ground(Departure,*)" />
      </condition>
      <effect prob="1">
        <set var="current_step" value="Date" />
      </effect>
    </case>
    <case>
      <condition>
        <if var="a_m" value="Ground(Date,*)" />
      </condition>
      <effect prob="1">
        <set var="current_step" value="Return" />
      </effect>
    </case>
```

3.3 实战案例：航空订票机器人语料库（自然语言生成）

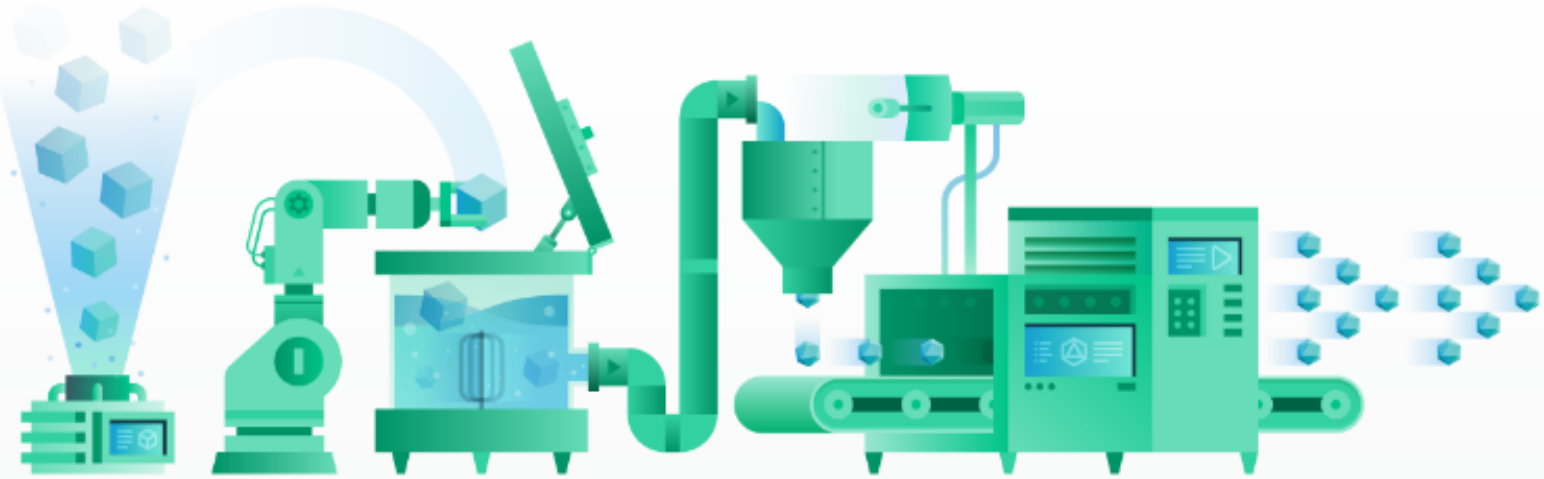
```
<model trigger="current_step">
  <!--Model producing new system utterances when the current step is moved
    to a new step. -->

  <!-- Asks for the destination -->
  <rule>
    <case>
      <condition>
        <if var="current_step" value="Destination" />
      </condition>
      <effect util="1">
        <set var="u_m" value="请问您要去那个城市？" />
      </effect>
    </case>
  </rule>

  <!-- Asks for the departure -->
  <rule>
    <case>
      <condition>
        <if var="current_step" value="Departure" />
      </condition>
      <effect util="1">
        <set var="u_m" value="请问您打算从那个城市出发？" />
      </effect>
    </case>
  </rule>
```

第四部分

第四部分 任务式智能对话机器人

- 任务式智能对话机器人基础
- 任务式智能对话机器人案例
- 任务式智能对话机器人实战
- 任务式智能对话机器人源码

4.1 源码安装部署：Gradle官网



Accelerate developer productivity

From mobile apps to microservices, from small startups to big enterprises,
Gradle helps teams build, automate and deliver better software, faster.

1. Install Gradle → 2. Get Started Guides → 3. Free Training

4.1 源码安装部署：Gradle下载

Step 1. Download **the latest Gradle distribution**

The current Gradle release is version 4.5.1, released on 05 Feb 2018. The distribution zip file comes in two flavors:

- Binary-only (sha256)
- Complete, with docs and sources (sha256)

If in doubt, choose the binary-only version and browse docs and sources online.

Need to work with an older version? See the releases page.

Step 2. Unpack the distribution

Linux & MacOS users

Unzip the distribution zip file in the directory of your choosing, e.g.:

```
$ mkdir /opt/gradle
$ unzip -d /opt/gradle gradle-4.5.1-bin.zip
$ ls /opt/gradle/gradle-4.5.1
LICENSE  NOTICE  bin  getting-started.html  init.d  lib  media
```

4.1 源码安装部署：Gradle配置

Microsoft Windows users

Create a new directory `C:\Gradle` with **File Explorer**.

Open a second **File Explorer** window and go to the directory where the Gradle distribution was downloaded. Double-click the ZIP archive to expose the content. Drag the content folder `gradle-4.5.1` to your newly created `C:\Gradle` folder.

Alternatively you can unpack the Gradle distribution ZIP into `C:\Gradle` using an archiver tool of your choice.

Step 3. Configure your system environment

Linux & MacOS users

Configure your `PATH` environment variable to include the `bin` directory of the unzipped distribution, e.g.:

```
$ export PATH=$PATH:/opt/gradle/gradle-4.5.1/bin
```

4.1 源码安装部署：Gradle验证

Microsoft Windows users

In **File Explorer** right-click on the `This PC` (or `Computer`) icon, then click `Properties` -> `Advanced System Settings` -> `Environmental Variables`.

Under `System Variables` select `Path`, then click `Edit`. Add an entry for `C:\Gradle\gradle-4.5.1\bin`. Click OK to save.

Step 4. Verify your installation

Open a console (or a Windows command prompt) and run `gradle -v` to run gradle and display the version, e.g.:

```
$ gradle -v

------------------------------------------------------------
Gradle 4.5.1
------------------------------------------------------------
```

4.1 源码编译-OpenDial

3. Compiling OpenDial from source (*optional*)

In case you plan to work on OpenDial's source code, or if you have fetched the bleeding-edge source code from the github repository, you will need to recompile the source. Since version 1.4, OpenDial relies on the Gradle build framework (before 1.4, we used ant).

If you do not yet have Gradle on your machine, you first need to install it (on Mac OS X, you can install it via Brew, and a gradle package is also available on Ubuntu). Once this is done, simply go to the main directory and type:

```
gradle compile
```

This should compile the source code in a few seconds. You can verify that everything works correctly by running the unit tests for Opendial:

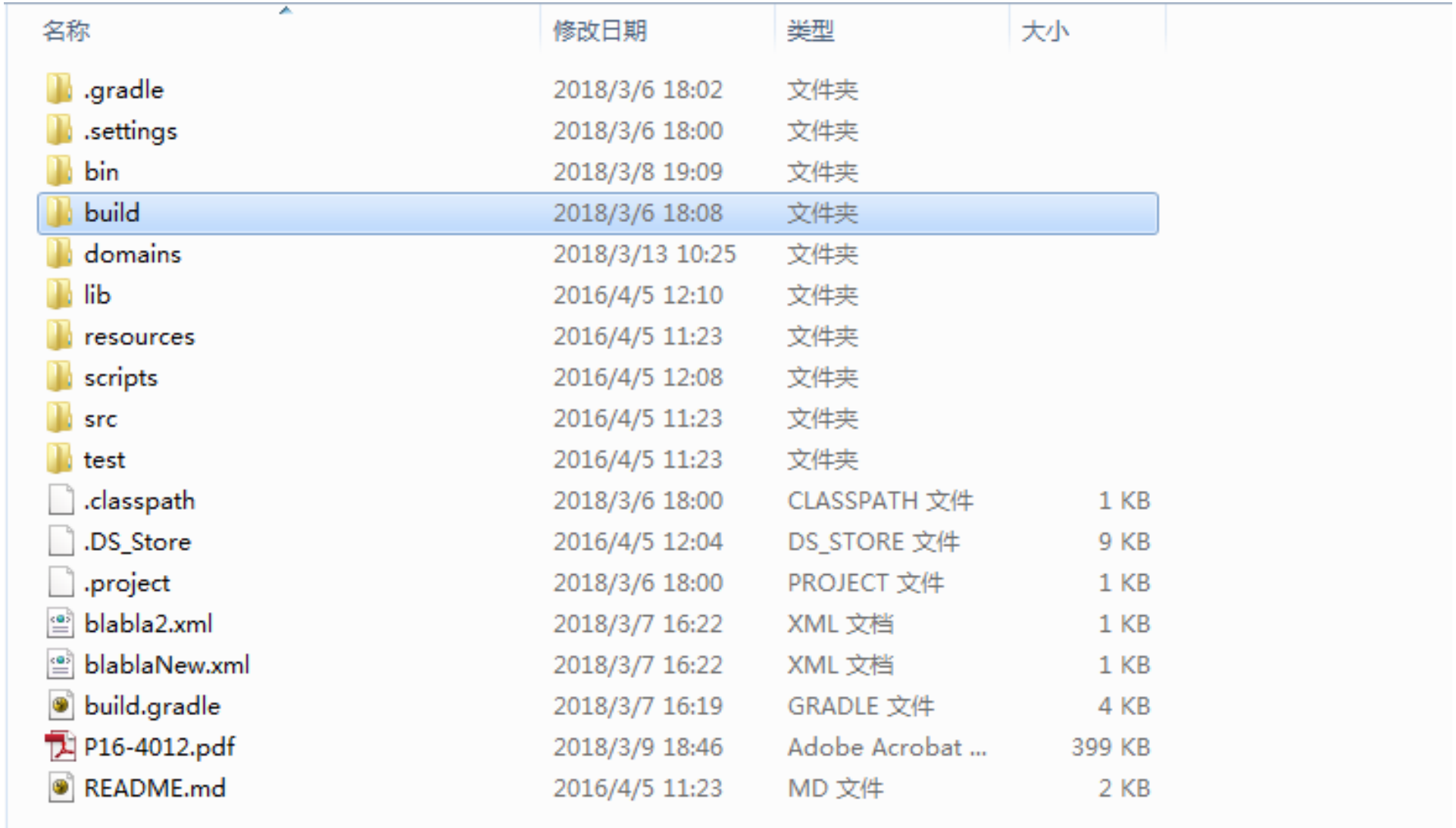
```
gradle test
```

Once the compilation is completed, you can run OpenDial through the scripts mentioned above (`./scripts/opendial` or `.\scripts\opendial.bat`).

4.1 源码编译-OpenDial-编译错误

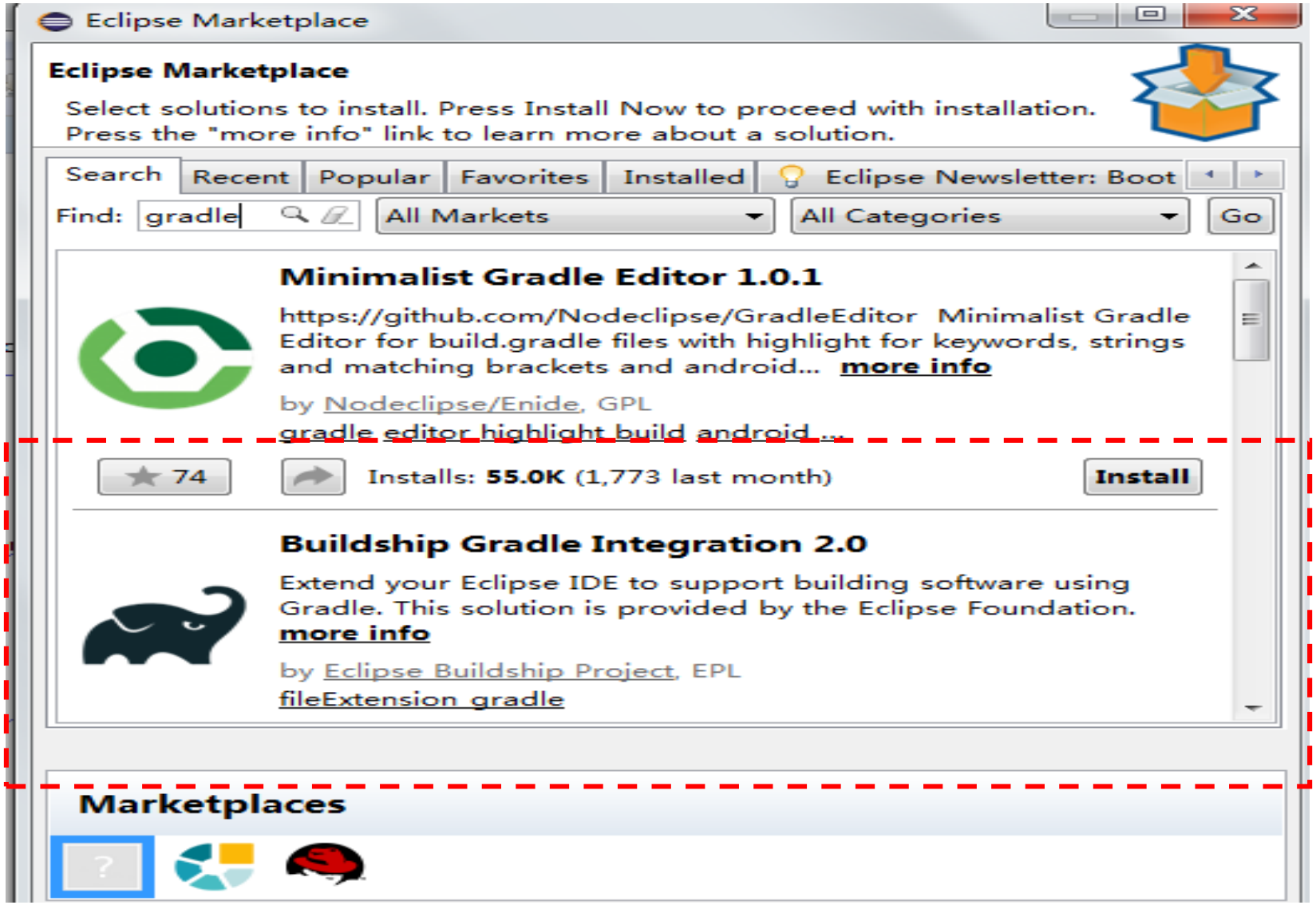
```
FAILURE: Build failed with an exception.
* What went wrong:
Execution failed for task ':compileJava'.
> Could not find tools.jar. Please check that C:\Program Files\Java\jre1.8.0_144
  contains a valid JDK installation.

* Try:
Run with --stacktrace option to get the stack trace. Run with --info or --debug
option to get more log output. Run with --scan to get full insights.

* Get more help at https://help.gradle.org

Deprecated Gradle features were used in this build, making it incompatible with
Gradle 5.0.
See https://docs.gradle.org/4.5.1/userguide/command_line_interface.html#sec:comm
and_line_warnings
BUILD FAILED in 2m 8s
1 actionable task: 1 executed
```

4.1 源码编译-OpenDial-输出结果

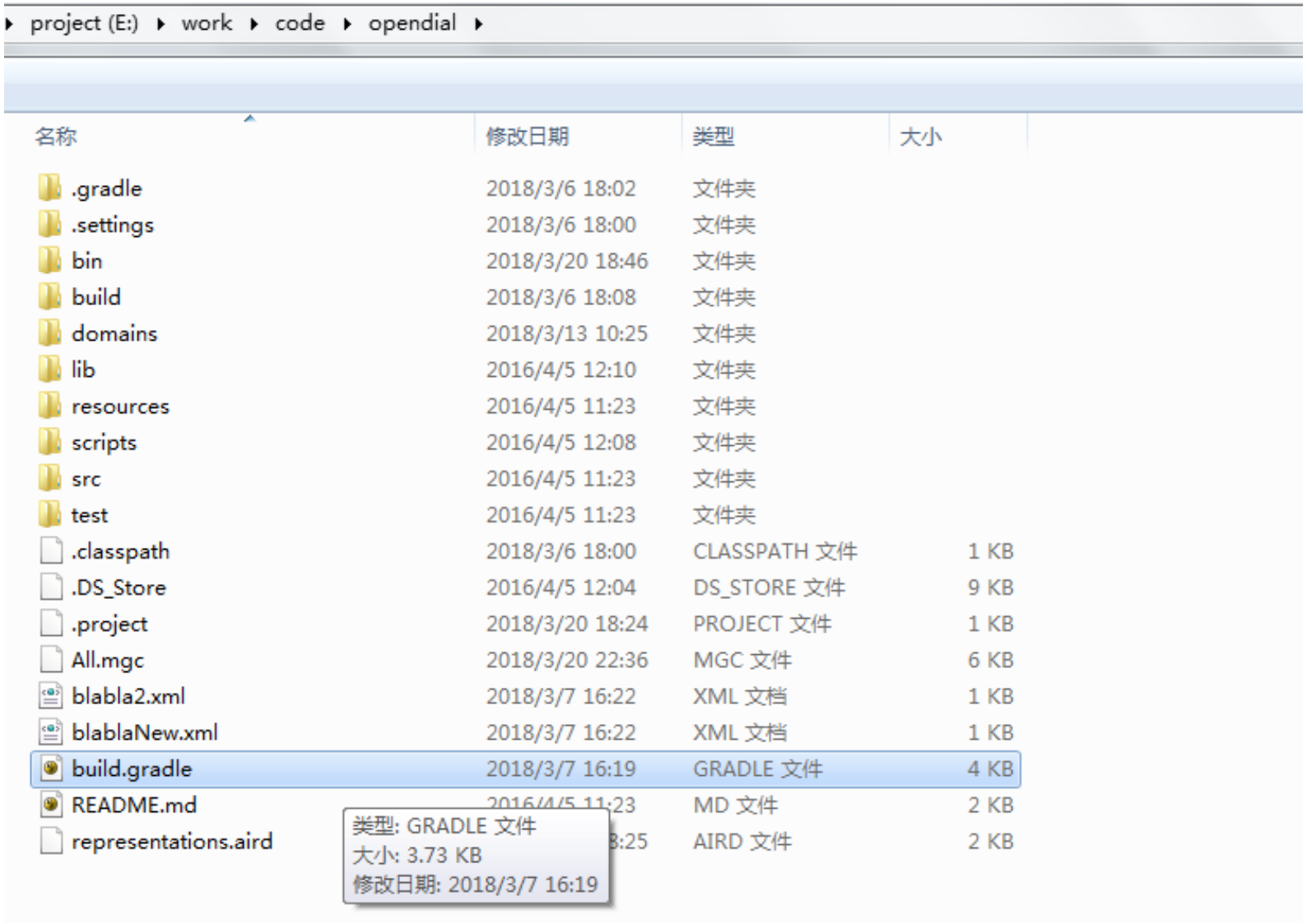


名称	修改日期	类型	大小
.gradle	2018/3/6 18:02	文件夹	
.settings	2018/3/6 18:00	文件夹	
bin	2018/3/8 19:09	文件夹	
build	2018/3/6 18:08	文件夹	
domains	2018/3/13 10:25	文件夹	
lib	2016/4/5 12:10	文件夹	
resources	2016/4/5 11:23	文件夹	
scripts	2016/4/5 12:08	文件夹	
src	2016/4/5 11:23	文件夹	
test	2016/4/5 11:23	文件夹	
.classpath	2018/3/6 18:00	CLASSPATH 文件	1 KB
.DS_Store	2016/4/5 12:04	DS_STORE 文件	9 KB
.project	2018/3/6 18:00	PROJECT 文件	1 KB
blabla2.xml	2018/3/7 16:22	XML 文档	1 KB
blablaNew.xml	2018/3/7 16:22	XML 文档	1 KB
build.gradle	2018/3/7 16:19	GRADLE 文件	4 KB
P16-4012.pdf	2018/3/9 18:46	Adobe Acrobat ...	399 KB
README.md	2016/4/5 11:23	MD 文件	2 KB

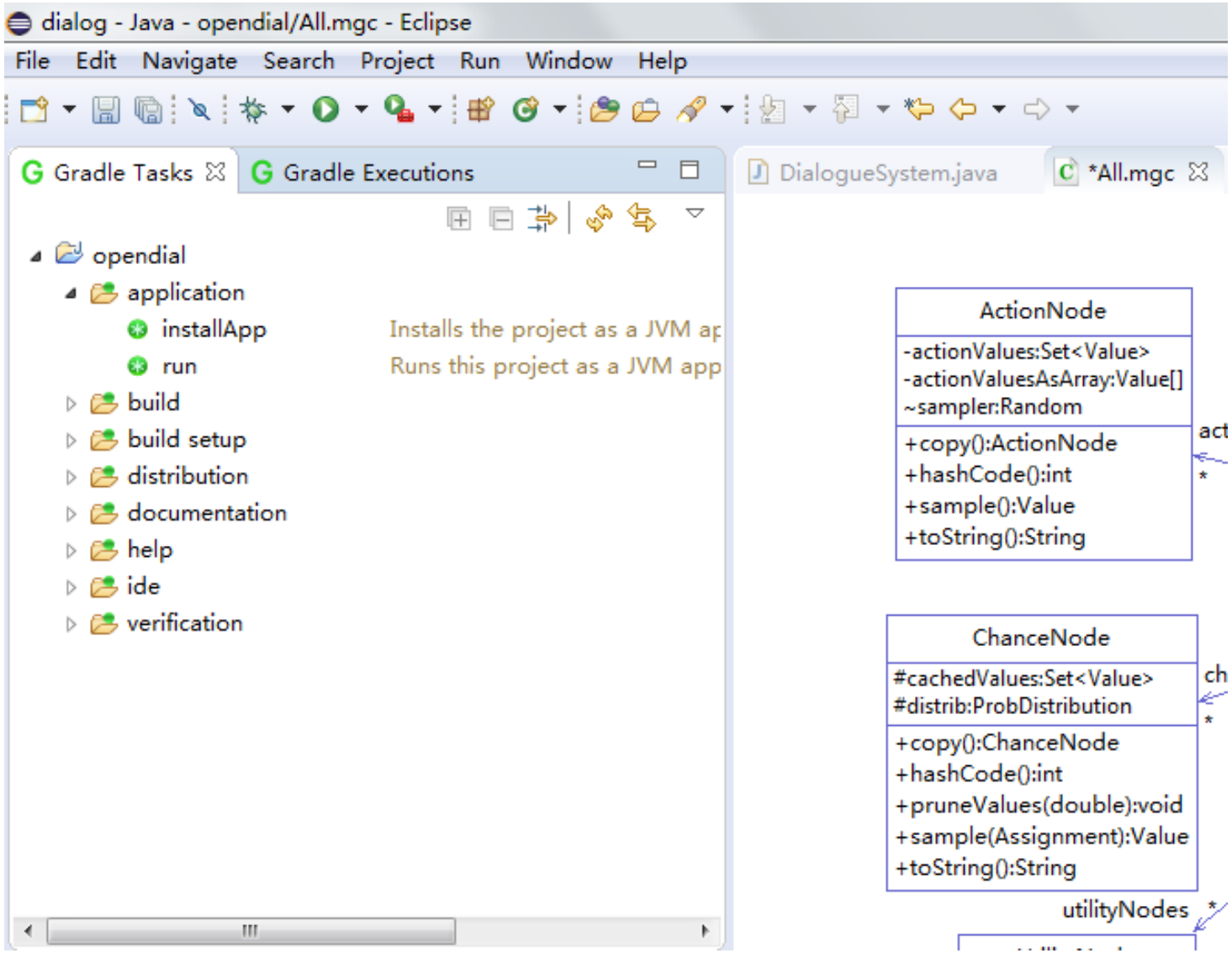
4.2 Eclipse与Opendia的集成：安装



4.2 Eclipse与Opendia的集成：打开



4.2 Eclipse与Opendia的集成：编译执行



4.3 Eclipse与ModelGoon集成

1、ModelGoon介绍

ModelGoon 是一个 Eclipse插件，能将Eclipse中现有的java类生成类图，可以进行Java 包的依赖分析，基于UML图进行模型设计，以及逆向工程（即从已有源代码生成类图）。

2、ModelGoon下载与安装

2.1 下载

本示例的环境为：Windows 7_X64, Eclipse Juno， JDK1.7, JRE1.7。待安装的ModelGoon版本为：ModelGoon-4.4.1-site.zip。

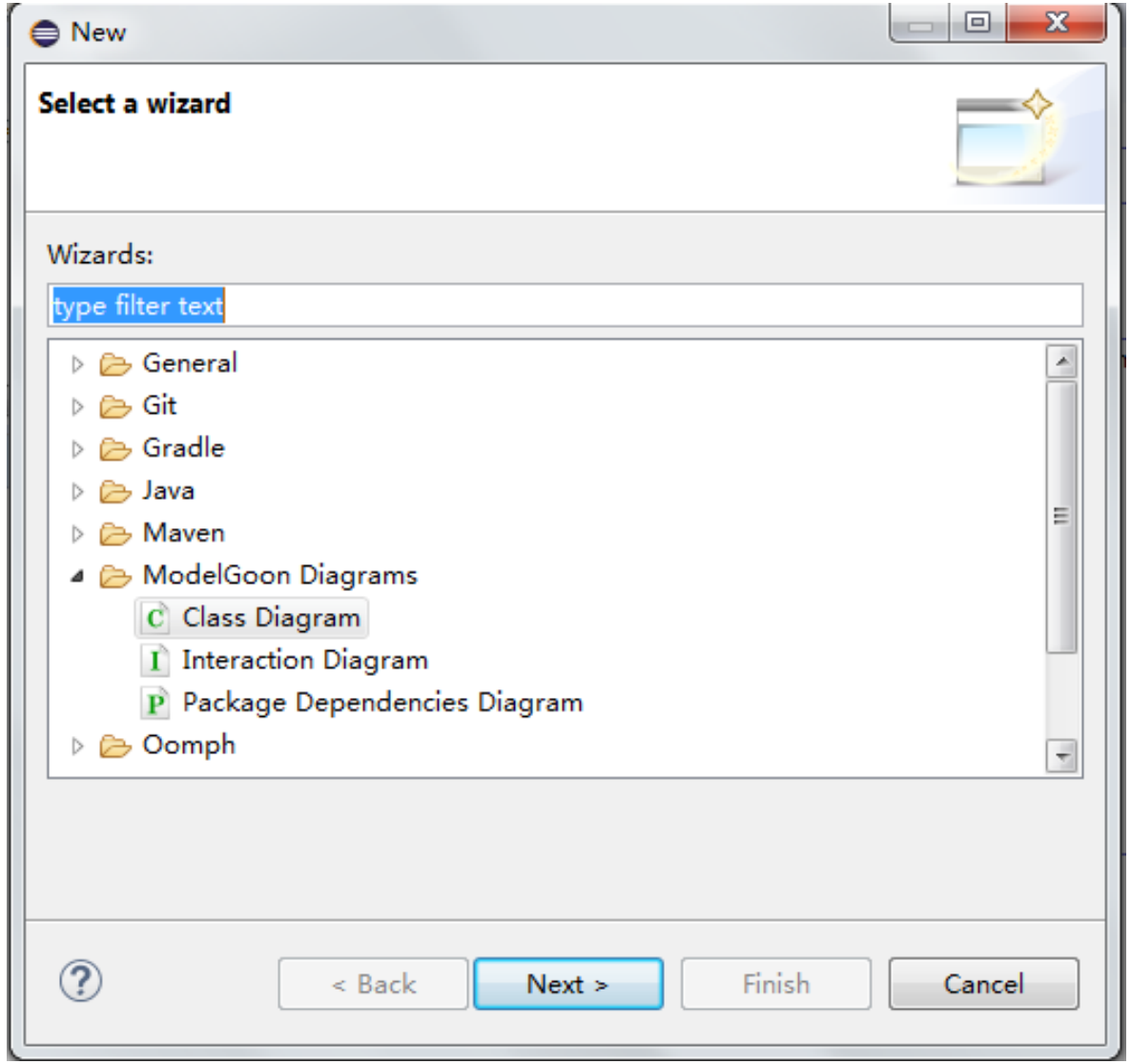
2.2 安装

打开eclipse，选择help-->install new software。点击work with-->Add-->Archive，选择已经下载的ModelGoon-4.4.1-site.zip，一路next或者accept，最后选择finish完成安装。(有的时候安装过程会比较长，还可能出现类似卡死的现象，长时间没有任何进度。这时候不要取消，耐心等待就好)。该插件不能通过help-->EclipseMarketplace进行安装。

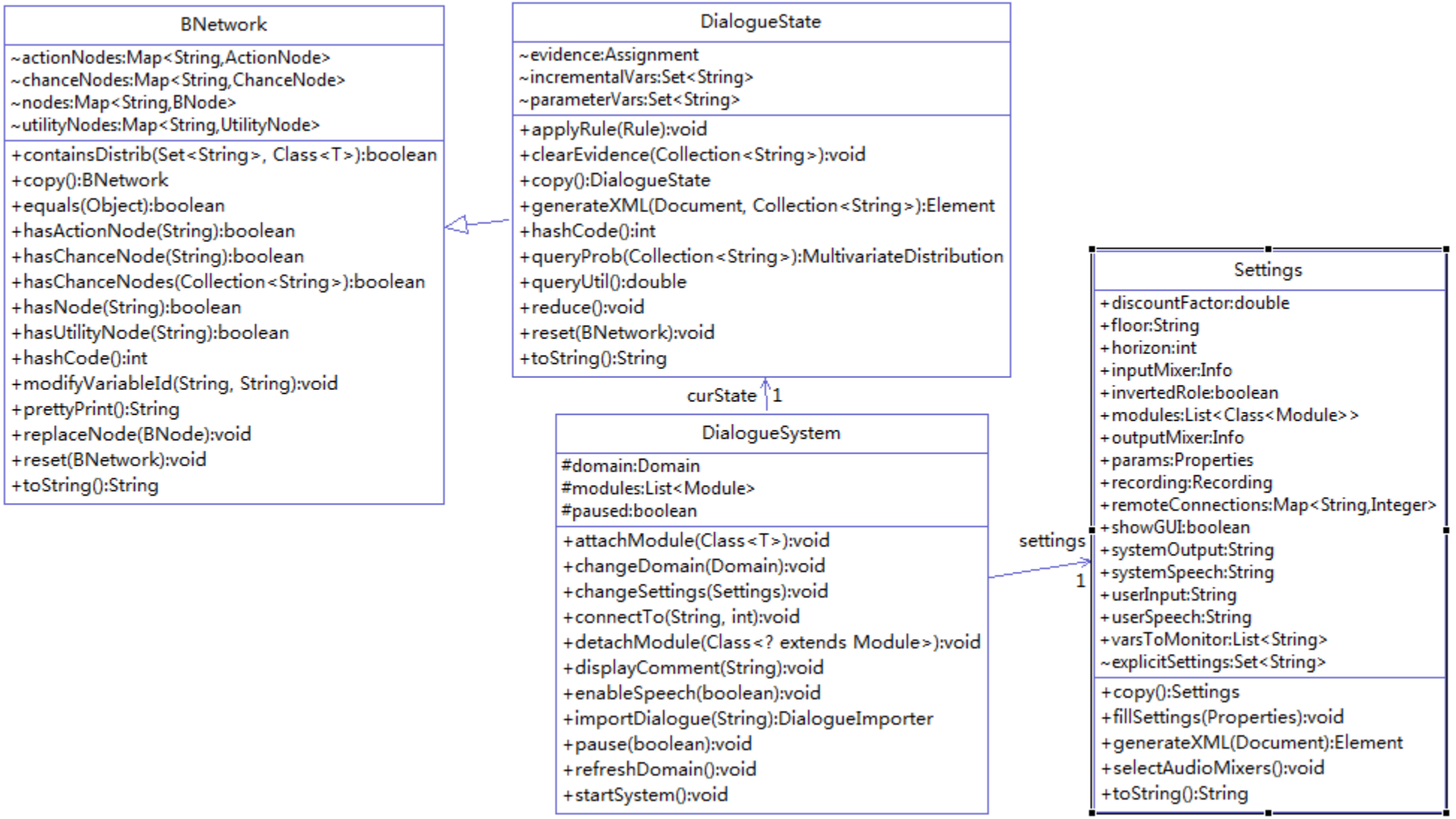
3、ModelGoon使用

ModelGoon安装成功后，重新启动eclipse。在eclipse中，选择已经打开的Java或者Android工程。在合适的文件夹或者直接在工程根目录中，File-->new-->other-->ModelGoon Diagrams，如下图所示：

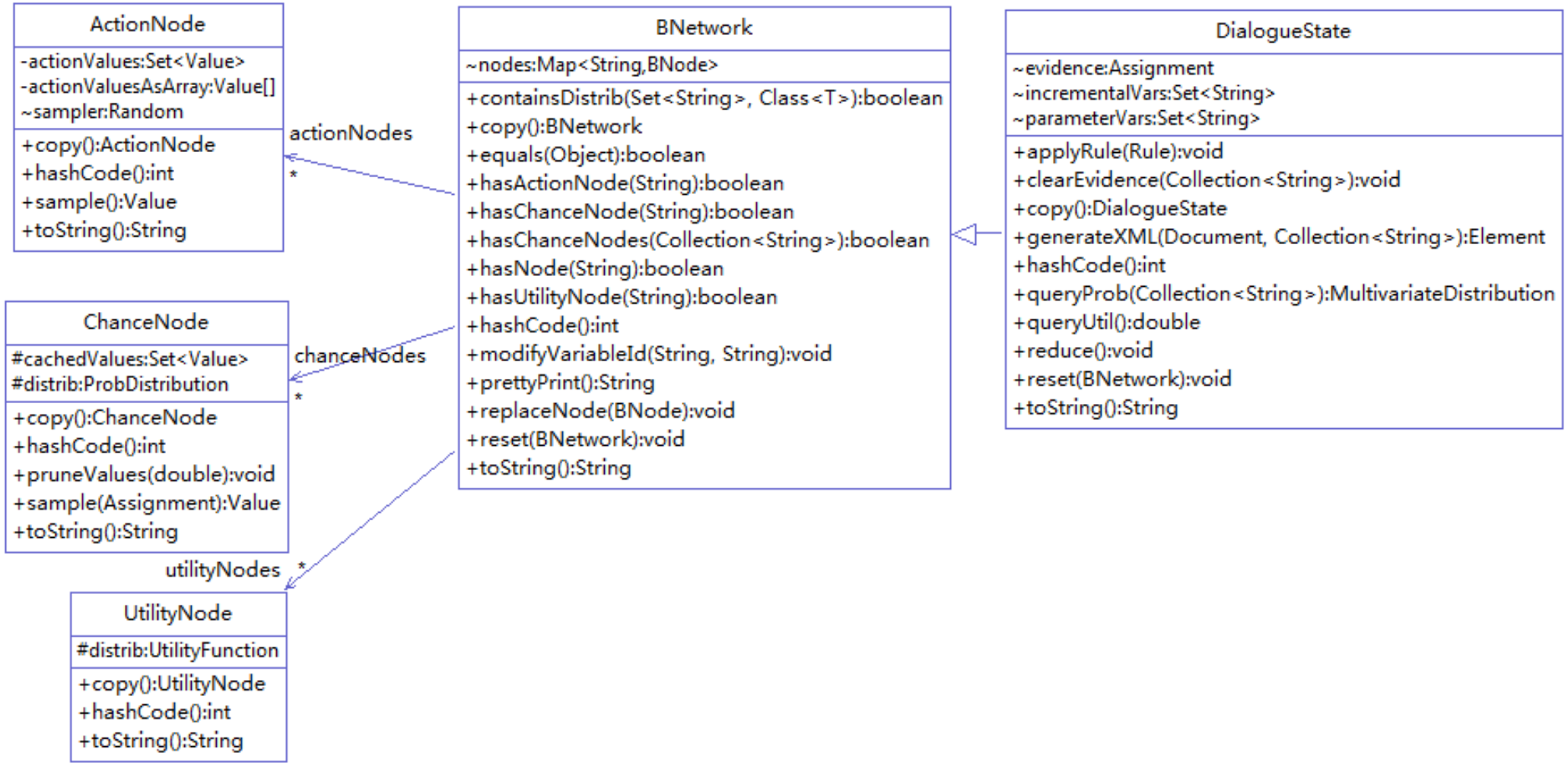
4.3 Eclipse与ModelGoon集成



4.4 Opendial源码框架解析-核心类



4.4 Opendial源码框架解析-核心类



4.4 Opendial源码框架解析-Main（ ）

```
842     // ===================================
843     // MAIN METHOD
844     // ===================================
845
846     /**
847      * Starts the dialogue system. The content of the args array is ignored.
848      * Command-line parameters can however be specified through system properties via
849      * the -D flag. All parameters are optional.
850      *
851      * <p>
852      * Some of the possible properties are:
853      * <ul>
854      * <li>-Ddomain=path/to/domain/file: dialogue domain file
855      * <li>-Ddialogue=path/to/recorded/dialogue: dialogue file to import
856      * <li>-Dsimulator=path/to/simulator/file: domain file for the simulator
857      * <li>--Dgui=true or false: activates or deactives the GUI
858      * </ul>
859      *
860      * @param args is ignored.
861      * @throws IOException
862      */
863     public static void main(String[] args) throws IOException {
864
865         DialogueSystem system = new DialogueSystem();
866         String domainFile = System.getProperty("domain");
867         String dialogueFile = System.getProperty("dialogue");
868         String simulatorFile = System.getProperty("simulator");
869
870         system.getSettings().fillSettings(System.getProperties());
```

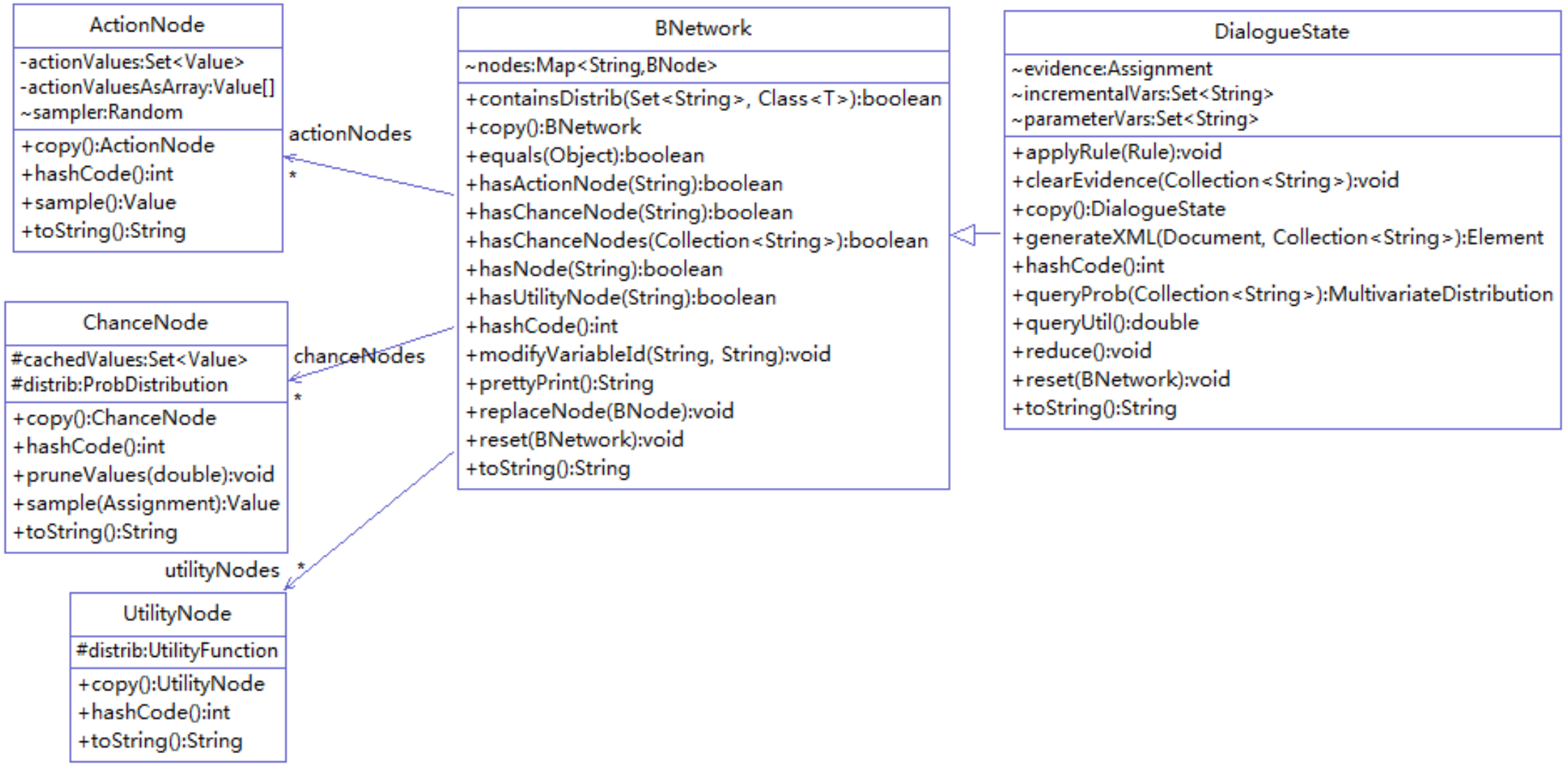
4.4 Opendial源码框架解析-Main（ ）

```
869         //1.Domain
870         if (domainFile != null) {
871             Domain domain;
872             try {
873                 domain = XMLDomainReader.extractDomain(domainFile);
874                 log.info("Domain from " + domainFile + " successfully extracted");
875             }
876             catch (RuntimeException e) {
877                 system.displayComment("Cannot load domain: " + e);
878                 e.printStackTrace();
879                 domain = XMLDomainReader.extractEmptyDomain(domainFile);
880             }
881             system.changeDomain(domain);
882         }
883         //2.Dialogue history
884         if (dialogueFile != null) {
885             system.importDialogue(dialogueFile);
886         }
887         //3. simulator
888         if (simulatorFile != null) {
889             Simulator simulator = new Simulator(system,
890                     XMLDomainReader.extractDomain(simulatorFile));
891             log.info("Simulator with domain " + simulatorFile
892                     + " successfully extracted");
893             system.attachModule(simulator);
894         }
895         Settings settings = system.getSettings();
896         system.changeSettings(settings);
897         //4.文本接口-单独只是文本，没有界面
```

4.4 Opendial源码框架解析-Main（ ）

```
        //4.文本接口-单独只是文本，没有界面
        if (!settings.showGUI) {
            system.attachModule(new TextOnlyInterface(system));
        }
        system.startSystem();
        log.info("Dialogue system started!");
    }

}
```

4.5.1 Opendial源码框架解析：对话状态表示



4.5.1 Opendial源码框架解析：对话状态表示

```
44 /**
45  * Representation of a Bayesian Network augmented with value and action nodes. The
46  * network is simply defined as a set of nodes connected with each other.
47  * 带有值和动作节点的贝叶斯网络的表示。网络简单地定义为一组互相连接的节点
48  * @author Pierre Lison (plison@ifi.uio.no)
49  *
50  */
51 public class BNetwork {
52     // logger
53     final static Logger log = Logger.getLogger("OpenDial");
54
55     // the set of nodes for the network
56     Map<String, BNode> nodes;
57
58     // the chance nodes
59     Map<String, ChanceNode> chanceNodes;
60
61     // the utility nodes
62     Map<String, UtilityNode> utilityNodes;
63
64     // the action nodes
65     Map<String, ActionNode> actionNodes;
```

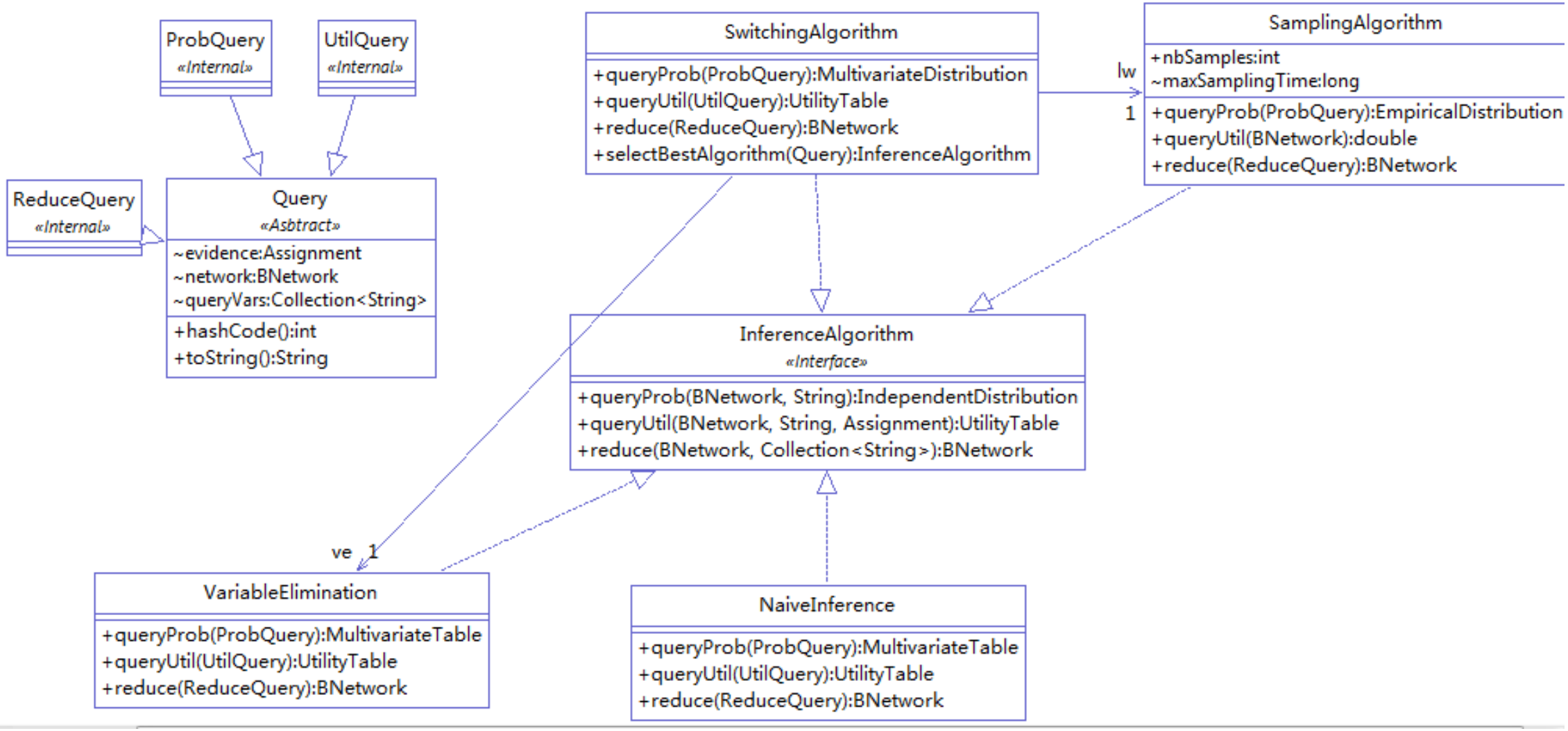
4.5.1 Opendial源码框架解析：对话状态表示

```
101     /**
102      * Adds a new node to the network. Note: if the node already exists, it is better
103      * to use the "replaceNode" method, to avoid warning messages.
104      * 添加节点到贝叶斯网络：基于节点类型，判定应该添加到那个
105      * @param node the node to add
106      */
107     public void addNode(BNode node) {
108         if (nodes.containsKey(node.getId())) {
109             log.warning("network already contains a node with identifier "
110                     + node.getId());
111         }
112         nodes.put(node.getId(), node);
113         node.setNetwork(this);
114         //机会节点
115         // adding the node in the type-specific collections
116         if (node instanceof ChanceNode) {
117             chanceNodes.put(node.getId(), (ChanceNode) node);
118         }
119         //功能节点:
120         else if (node instanceof UtilityNode) {
121             utilityNodes.put(node.getId(), (UtilityNode) node);
122         }
123         //行为节点/决策节点:
124         else if (node instanceof ActionNode) {
125             actionNodes.put(node.getId(), (ActionNode) node);
126         }
```

4.5.1 Opendial源码框架解析：对话状态表示

```
220      * Adds a new node to the dialogue state with the distribution provided as
221      * argument.
222      * 添加节点到对话状态管理器，根据节点类型进行判断吗？--No，此处的节点都是人的行为，属于机会节点
223      * @param distrib the distribution to include
224      */
225     public void addToState(ProbDistribution distrib) {
226         String variable = distrib.getVariable() + "'";
227         setAsCommitted(variable);
228         distrib.modifyVariableId(distrib.getVariable(), variable);
229         ChanceNode newNode = new ChanceNode(variable, distrib);
230 
231         if (hasNode(variable)) {
232             BNode toRemove = getNode(variable);
233             removeNodes(toRemove.getDescendantIds());
234             removeNode(toRemove.getId());
235         }
236         for (String inputVar : distrib.getInputVariables()) {
237             if (hasChanceNode(inputVar)) {
238                 newNode.addInputNode(getChanceNode(inputVar));
239             }
240         }
241         //01-添加新的节点
242         addNode(newNode);
243         //02-连接到预测
244         connectToPredictions(newNode);
```

4.5.2 Opendial源码框架解析：对话行为推理



4.5.2 Opendial源码框架解析：对话行为推理

```
/**
 * Switching algorithms that alternates between an exact algorithm (variable
 * elimination) and an approximate algorithm (likelihood weighting) depending on the
 * query.
 * 切换算法：根据查询的精确算法（变量消除）和近似算法（似然加权）之间的切换算法。
 * <p>
 * The switching mechanism is defined via two thresholds:
 * 阈值：
 * <ul>
 * <li>one threshold on the maximum branching factor of the network
 * <li>one threshold on the maximum number of combination of values in a node factor
 * </ul>
 * 网络最大分枝因子的一个阈值
 * 节点因子中最大值组合的一个阈值
 * <p>
 * If one of these threshold is exceeded or if the Bayesian network contains a
 * continuous distribution, the selected algorithm will be likelihood weighting.
 * Variable elimination is selected in the remaining cases.
 *
 * @author Pierre Lison (plison@ifi.uio.no)
 */
public class SwitchingAlgorithm implements InferenceAlgorithm {
```

4.5.2 Opendial源码框架解析：对话行为推理

```
426     /**
427      * Returns the probability distribution corresponding to the values of the state
428      * variables provided as argument.
429      * 返回状态值对应的概率分布
430      * @param variables the variable labels to query
431      * @return the corresponding probability distribution
432      */
433     public MultivariateDistribution queryProb(Collection<String> variables) {
434 
435         if (!getNodeIds().containsAll(variables)) {
436             log.warning(variables + " not contained in " + getNodeIds());
437         }
438         // else, perform the inference operation
439         try {
440             return new SwitchingAlgorithm().queryProb(this, variables, evidence);
441         }
442 
443         // if everything fails, returns an empty table
444         catch (Exception e) {
445             log.warning("cannot perform inference: " + e);
446             e.printStackTrace();
447             return new MultivariateTable(Assignment.createDefault());
448         }
449     }
```

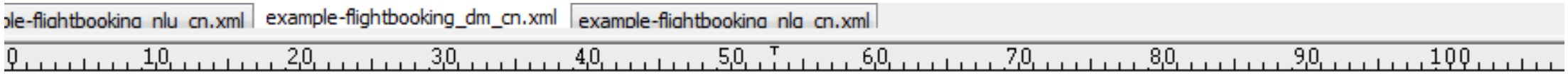
4.5.2 Opendial源码框架解析：对话行为推理

```
127    public InferenceAlgorithm selectBestAlgorithm(Query query) {
128
129        for (BNode node : query.getFilteredSortedNodes()) {
130            if (node.getInputNodeIds().size() > MAX_BRANCHING_FACTOR) {
131                return lw;
132            }
133            if (node instanceof ChanceNode) {
134                if (((ChanceNode) node)
135                        .getDistrib() instanceof ContinuousDistribution) {
136                    return lw;
137                }
138                int nbValues = ((ChanceNode) node).getNbValues();
139                for (ChanceNode i : node.getInputNodes(ChanceNode.class)) {
140                    nbValues *= i.getNbValues();
141                }
142                if (nbValues > MAX_NBVALUES) {
143                    return lw;
144                }
145            }
146        }
147        return ve;
148    }
```

4.6 平台框架拓展：航空订票拓展实例

```
<case>
  <condition>
    <if var="a_m" value="Ground(NoReturn)" />
  </condition>
  <effect prob="1">
    <set var="ReturnDate" value="NoReturn" />
    <set var="a_m-prev" value="{a_m}" />
    <set var="a_m" value="FindOffer" />
  </effect>
</case>
<case>
  <condition>
    <if var="a_m" value="Ground(ReturnDate,*)" />
  </condition>
  <effect prob="1">
    <set var="a_m-prev" value="{a_m}" />
    <set var="a_m" value="FindOffer" />
  </effect>
</case>
```

4.6 平台框架拓展：航空订票拓展实例




```
<!-- If the external module produces the system action MakeOffer(particular price),
registers the price in a separate variable and moves the current step. -->
<rule>
  <case>
    <condition>
      <if var="a_m" value="MakeOffer({Price})" />
    </condition>
    <effect>
      <set var="TotalCost" value="{Price}" />
      <set var="current_step" value="MakeOffer" />
    </effect>
  </case>
<!-- If the number of tickets is grounded, update the total cost by multiplying the
price with the number of tickets. -->
  <case>
    <condition>
      <if var="a_m" value="Ground(NbTickets,*)" />
    </condition>
    <effect>
      <set var="TotalCost" value="{TotalCost}*{NbTickets}" />
    </effect>
  </case>
</rule>
```

4.6 平台框架拓展：航空订票拓展实例

FlightBookingExample.java

opendial ▸ src ▸ opendial.modules.examples ▸ FlightBookingExample ▸

```
// Copyright (C) 2011-2015 Pierre Lison (plison@ifi.uio.no)

package opendial.modules.examples;

import java.util.logging.*;

/**
 * Example of simple external module used for the flight-booking dialogue domain. The
 * module monitors for two particular values for the system action:
 * <ol>
 * <li>"FindOffer" checks the (faked) price of the user order and returns
 * MakeOffer(price)
 * <li>"Book" simulates the booking of the user order.
 * </ol>
 *
 * @author Pierre Lison (plison@ifi.uio.no)
 */
public class FlightBookingExample implements Module {

    // logger
    public final static Logger log = Logger.getLogger("OpenDial");

    // the dialogue system
    DialogueSystem system;
```

4.6 平台框架拓展：航空订票拓展实例

```
67      @Override
68      public void start() {
69          paused = false;
70      }
71
72      /**
73       * Checks whether the updated variables contains the system action and (if yes)
74       * whether the system action value is "FindOffer" or "Book". If the value is
75       * "FindOffer", checks the price of the order (faked here to 179 or 299 EUR) and
76       * adds the new action "MakeOffer(price)" to the dialogue state. If the value is
77       * "Book", simply write down the order on the system output.
78       *
79       * @param state the current dialogue state
80       * @param updatedVars the updated variables in the state
81       * 自定义处理函数的位置：获取变量，并且执行对应的查询操作
82       */
83      @Override
84      public void trigger(DialogueState state, Collection<String> updatedVars) {
85          if (updatedVars.contains("a_m") && state.hasChanceNode("a_m")) {
86              String action = state.queryProb("a_m").getBest().toString();
87
88              if (action.equals("FindOffer")) {
89                  String returndate =
90                          state.queryProb("ReturnDate").getBest().toString();
91
```

4.6 平台框架拓展：航空订票拓展实例

FlightBookingExample.java

opendial ▸ src ▸ opendial.modules.examples ▸ FlightBookingExample ▸ trigger(DialogueState, Collection<String>) : void

```
95                  // system.
96                  int price = (returndate.equals("NoReturn")) ? 179 : 299;
97                  String newAction = "MakeOffer(" + price + ")";
98                  system.addContent("a_m", newAction);
99              }
100             else if (action.equals("Book")) {
101
102                 String departure = state.queryProb("Departure").getBest().toString();
103                 String destination =
104                         state.queryProb("Destination").getBest().toString();
105                 String date = state.queryProb("Date").getBest().toString();
106                 String returndate =
107                         state.queryProb("ReturnDate").getBest().toString();
108                 String nbtickets = state.queryProb("NbTickets").getBest().toString();
109
110                 // In a real system, the system database should be modified here
111                 // to
112                 // actually perform the booking. Here, we just print a small
113                 // message.
114                 String info = "Booked " + nbtickets + " tickets from " + departure
115                         + " to " + destination + " on " + date
116                         + ((returndate.equals("NoReturn"))
117                                 ? " and return on " + returndate : "");
118                 log.fine(info);
119             }
120         }
```

谢谢聆听 请多指教