



**UNIVERSIDAD DE SANTIAGO DE CHILE
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA EN INFORMÁTICA**

**Manual de Usuario
Chatbot**

Programado en Prolog

Gabriel Gaete L.

Santiago de Chile

1 - 2018

TABLA DE CONTENIDO

<i>CAPÍTULO 1. INTRODUCCIÓN AL CHATBOT</i>	<i>4</i>
<i>CAPÍTULO 2. COMPILACIÓN Y EJECUCIÓN.....</i>	<i>5</i>
<i>CAPÍTULO 3. FUNCIONALIDADES Y MODOS DE USO</i>	<i>6</i>
3.1 beginDialog(Chatbot, InputLog, Seed, OutputLog).....	6
3.2 sendMessage(Str, Chatbot, InputLog, Seed, OutputLog)	7
3.3 endDialog(Chatbot, InputLog, Seed, OutputLog)	8
3.4 logToStr(Log, StrRep)	9
3.5 test(User, Chatbot, InputLog, Seed, OutputLog)	10
<i>CAPÍTULO 4. EN CASO DE FALLOS</i>	<i>11</i>

TABLA DE FIGURAS

Figura 1 Relación usuario/chatbot.....	4
Figura 2 Flujo de conversación esperado por el chatbot.....	5
Figura 3 Permitir la visualización de listas	6
Figura 4 Ejemplo de uso beginDialog.	7
Figura 5 Intercambio de mensajes entre usuario y chatbot.....	7
Figura 6 Especificación de una capital regional a la cual viajar	8
Figura 7 Uso de endDialog	8
Figura 8 Ejemplo predicado rate	9
Figura 9 Predicado writeLog.....	9
Figura 10 Ejemplo de conversación simulada mediante predicado test	10
Figura 11 Muestra de múltiples usuarios funcionalidad test.....	11
Figura 12 Error de sintaxis.....	11

CAPÍTULO 1. INTRODUCCIÓN AL CHATBOT

Un chatbot es un programa que permite simular una conversación con una persona, entregando respuestas automáticas a entradas hechas por un usuario. Estos son utilizados por diferentes marcas y compañías para obtener información, reservar algo, o comprar un producto, entre muchas otras aplicaciones. El chatbot del presente manual permite hacer una compra de pasajes hacia capitales regionales de Chile, asumiendo que el usuario se encuentra en Santiago.

El presente manual pretende explicar cómo utilizar de manera efectiva el chatbot desarrollado en Prolog, a través de ejemplos del programa funcionando en un sistema operativo MacOSX, aunque también es posible de compilar y posteriormente ejecutar en otro entorno, ya sea Windows o Linux.

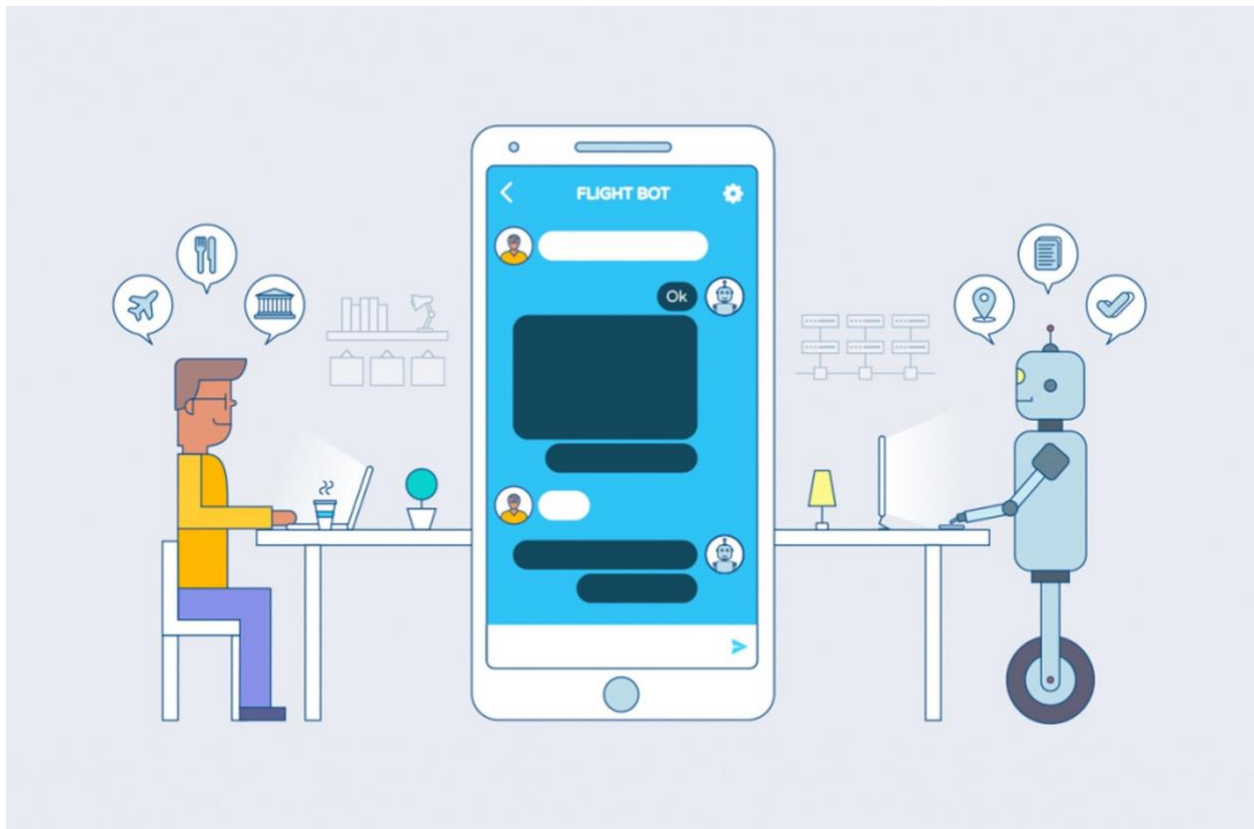


Figura 1 Relación usuario/chatbot (Fuente: <http://www.tci.net.pe/sera-2018-ano-los-chatbots/>)

Para la correcta utilización de este programa, se debe conocer cómo es el flujo conversacional esperado por el bot frente al usuario, por lo que se recomienda guiarse por el siguiente diagrama para evitar posibles errores en el flujo de la conversación.

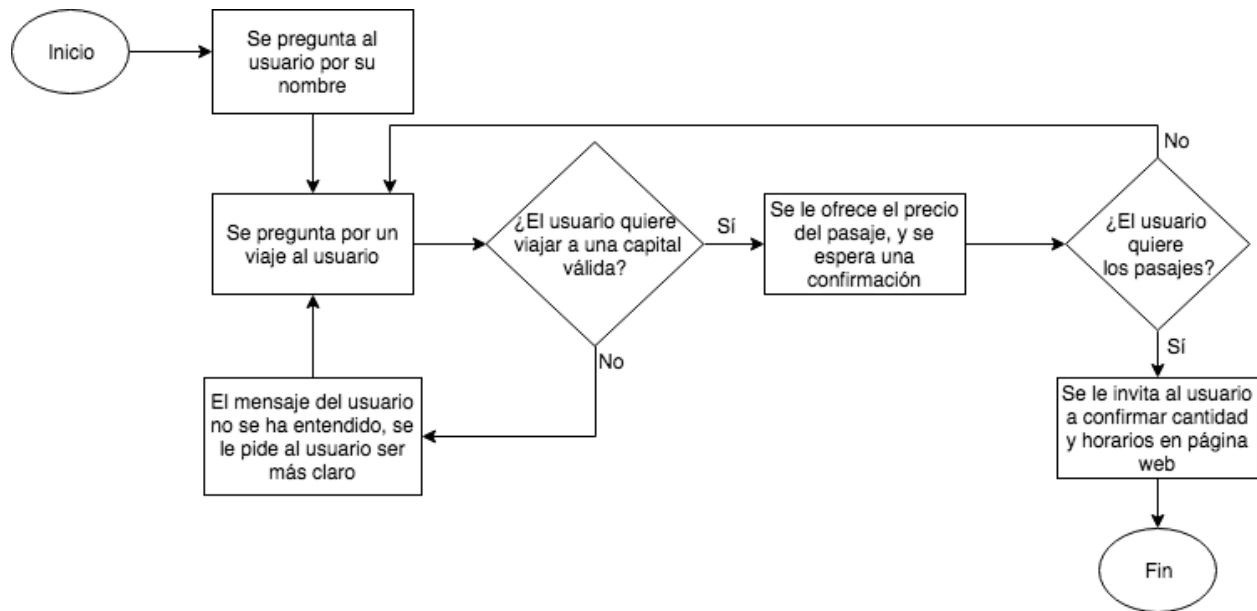


Figura 2 Flujo de conversación esperado por el chatbot

CAPÍTULO 2. COMPILACIÓN Y EJECUCIÓN

Para ejecutar, primero es necesario utilizar el compilador **swipl**, ya sea para Windows o Linux, el comando es el siguiente:

```
X swipl -S chatbot_19753546_GaeteLucero.pl
```

Este comando debe ingresarse a través de la terminal, estando ubicados en el mismo directorio que el código fuente, es decir, la carpeta “*chatbot*” que acompaña a este documento.

Al utilizar este comando, no solamente se compila el código fuente, sino que también queda abierto un intérprete, mediante el cual se pueden realizar consultas sobre las funcionalidades del código.

CAPÍTULO 3. FUNCIONALIDADES Y MODOS DE USO

Iniciado el programa, se despliega una suerte de consola con la cual se puede interactuar, permitiendo ejecutar desde aquí todas las funcionalidades que posee el chatbot.

NOTA IMPORTANTE: ANTES DE REALIZAR LAS CONSULTAS EN EL INTÉRPRETE, REALIZAR LA SIGUIENTE OPERACIÓN.



```
chatbot — swipl -S chatbot_19753546_GaeteLucero.pl — swipl — swipl -S cha...
chatbot git:(master) * swipl -S chatbot_19753546_GaeteLucero.pl
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- true ; true.
true [write]
true .
```

Figura 3 Permitir la visualización de listas

El paso realizado en la Figura 3, consiste en hacer una consulta de **true ; true**. En el momento que se entrega el primer resultado “true”, se debe presionar la tecla **w**, posteriormente, se ingresa un **.** (punto), para dejar de mostrar resultados. Esto “activa” la opción de **[write]**, como se muestra en la Figura 3, permitiendo que *Prolog* sea capaz de mostrar listas en su totalidad, algo totalmente necesario para ejecutar las consultas que entreguen como resultado listas con más de 10 elementos.

3.1 beginDialog(Chatbot, InputLog, Seed, OutputLog)

El predicado `beginDialog` permite iniciar la conversación entre el usuario y el Chatbot. Para esta funcionalidad, solo se debe especificar el valor de la semilla, el resto de valores no son necesarios, ya que el programa es capaz de asumir internamente valores en caso de no ser precisados por el usuario, como se muestra en la siguiente figura.

```
chatbot — swipl -S chatbot_19753546_GaeteLucero.pl — swipl — swipl -S chatbot_19753546_GaeteLucero.pl — 111...
?- beginDialog(Chatbot, InputLog, 40, OutputLog).
Chatbot = [{"Hola, mi nombre es Bot, y estoy aquí para ayudarlo a seleccionar un destino. ¿Me podría decir su nombre?", "Hola, mi nombre es Bot, y si quieres viajar, conmigo debes hablar. ¿Cómo debo llamarte?"}, {"cuéntame, ¿a dónde quieres viajar? Recuerda que por el momento sólo ofrecemos viajes a capitales regionales del país.", "¿a qué capital regional deseas viajar? Puedes hacerlo a cualquier región de Chile. Yo te recomiendo el norte.", "y bueno, ¿a qué capital regional te gustaría ir? El sur es hermoso en toda época del año."}, [{"0", "0"}], [{"¿A qué ciudad entonces te gustaría ir?", "No hay problema, puedes elegir un nuevo destino"}, [{"¡Perfecto! Ahora, para confirmar pasajes, debe ingresar a nuestro sitio web.", "Bien, ahora para confirmar la cantidad y la fecha de los pasajes, debe ingresar a nuestro sitio web"}, {"es un lugar precioso! Los pasajes hacia allá cuestan", "es ideal en esta época del año, no te arrepentirás. Viajar hacia allá cuesta"}, {"Disculpa, no he logrado entenderte del todo... ¿podrías ser un poco más claro?", "Perdón, pero no he entendido lo que me has dicho... ¿podrías ser un poco más claro?"}, {"Hasta luego, espero haber sido de ayuda en esta oportunidad.", "Hasta la próxima, espero haberte ayudado."}],
InputLog = [],
OutputLog = [{"[20/5/2018, 17:18]", "BeginDialog", "ID:", "0"}, {"[20/5/2018, 17:18]", "Bot:", "Hola, mi nombre es Bot, y si quieres viajar, conmigo debes hablar. ¿Cómo debo llamarte?"}].
```

Figura 4 Ejemplo de uso beginDialog.

Como se observa en la Figura 4, se determina la estructura Chatbot que hace válido al predicado, se asume que el InputLog está vacío en caso de no ser especificado como entrada, y se entrega en el OutputLog el mensaje de bienvenida al chat.

3.2 sendMessage(Str, Chatbot, InputLog, Seed, OutputLog)

El predicado sendMessage permite el envío de un mensaje por parte del usuario al chatbot. Este mensaje parte del supuesto que se conoce el flujo de conversación que hay detrás, por lo que se recomienda tener en cuenta el diagrama presentado en la Figura 2. Este predicado recibe los mismos argumentos que el anterior (**beginDialog**), salvo por el mensaje, el cual debe ser un string (texto entre comillas dobles). En la siguiente figura, se muestra un ejemplo de consulta utilizando este predicado. Nótese que el *InputLog* corresponde al OutputLog generado por la consulta realizada en la Figura 3, mientras que el Chatbot nuevamente no se especifica.

```
chatbot — swipl -S chatbot_19753546_GaeteLucero.pl — swipl — swipl -S chatbot_19753546_GaeteLucero.pl — 111...
?- sendMessage("Gabriel", Chatbot, [{"[20/5/2018, 17:18]", "BeginDialog", "ID:", "0"}, {"[20/5/2018, 17:18]", "Bot:", "Hola, mi nombre es Bot, y si quieres viajar, conmigo debes hablar. ¿Cómo debo llamarte?"}], 40, OutputLog).
Chatbot = [{"Hola, mi nombre es Bot, y estoy aquí para ayudarlo a seleccionar un destino. ¿Me podría decir su nombre?", "Hola, mi nombre es Bot, y si quieres viajar, conmigo debes hablar. ¿Cómo debo llamarte?"}, {"cuéntame, ¿a dónde quieres viajar? Recuerda que por el momento sólo ofrecemos viajes a capitales regionales del país.", "¿a qué capital regional deseas viajar? Puedes hacerlo a cualquier región de Chile. Yo te recomiendo el norte.", "y bueno, ¿a qué capital regional te gustaría ir? El sur es hermoso en toda época del año."}, [{"0", "0"}], [{"¿A qué ciudad entonces te gustaría ir?", "No hay problema, puedes elegir un nuevo destino"}, [{"¡Perfecto! Ahora, para confirmar pasajes, debe ingresar a nuestro sitio web.", "Bien, ahora para confirmar la cantidad y la fecha de los pasajes, debe ingresar a nuestro sitio web"}, {"es un lugar precioso! Los pasajes hacia allá cuestan", "es ideal en esta época del año, no te arrepentirás. Viajar hacia allá cuesta"}, {"Disculpa, no he logrado entenderte del todo... ¿podrías ser un poco más claro?", "Perdón, pero no he entendido lo que me has dicho... ¿podrías ser un poco más claro?"}, {"Hasta luego, espero haber sido de ayuda en esta oportunidad.", "Hasta la próxima, espero haberte ayudado."}],
OutputLog = [{"[20/5/2018, 17:18]", "BeginDialog", "ID:", "0"}, {"[20/5/2018, 17:18]", "Bot:", "Hola, mi nombre es Bot, y si quieres viajar, conmigo debes hablar. ¿Cómo debo llamarte?"}, {"[20/5/2018, 18:11]", "Usuario:", "Gabriel"}, {"[20/5/2018, 18:11]", "Bot:", "Gabriel ¿a qué capital regional deseas viajar? Puedes hacerlo a cualquier región de Chile. Yo te recomiendo el norte."}].
```

Figura 5 Intercambio de mensajes entre usuario y chatbot

Ahora, utilizando el outputLog de la Figura 5, la siguiente figura muestra un mensaje válido para indicar el viaje a una capital regional, respetando el capitalizado de la palabra, junto con su ortografía.

```
chatbot — swipl -S chatbot_19753546_GaeteLucero.pl — swipl — swipl -S chatbot_19753546_GaeteLucero.pl — 111...
?- sendMessage("tengo ganas de viajar a Valparaíso", Chatbot, [{"[20/5/2018, 17:18]", "BeginDialog", "ID:", "0"}], [{"[20/5/2018, 17:18]", "Bot:", "Hola, mi nombre es Bot, y si quieres viajar, conmigo debes hablar. ¿Cómo debo llamarte?"}, [{"[20/5/2018, 18:11]", "Usuario:", "Gabriel"}, [{"[20/5/2018, 18:11]", "Bot:", "Gabriel ¿a qué capital regional deseas viajar? Puedes hacerlo a cualquier región de Chile. Yo te recomiendo el norte."}], 40, OutputLog).
Chatbot = [{"[20/5/2018, 17:18]", "BeginDialog", "ID:", "0"}, [{"[20/5/2018, 17:18]", "Bot:", "Hola, mi nombre es Bot, y si quieres viajar, conmigo debes hablar. ¿Cómo debo llamarte?"}, [{"[20/5/2018, 18:11]", "Usuario:", "Gabriel"}, [{"[20/5/2018, 18:11]", "Bot:", "Gabriel ¿a qué capital regional deseas viajar? Puedes hacerlo a cualquier región de Chile. Yo te recomiendo el norte."}, [{"[20/5/2018, 18:17]", "Usuario:", "tengo ganas de viajar a Valparaíso"}, [{"[20/5/2018, 18:17]", "Bot:", "Valparaíso es ideal en esta época del año, no te arrepentirás. Viajar hacia allá cuesta $6.500 pesos ¿Desea confirmar su destino?"}], 40, OutputLog).
?-
```

Figura 6 Especificación de una capital regional a la cual viajar

3.3 endDialog(Chatbot, InputLog, Seed, OutputLog)

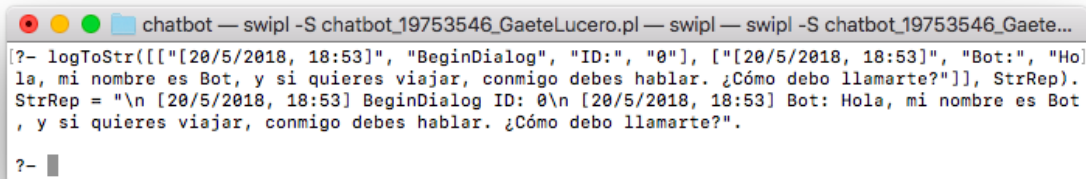
La consulta sobre este predicado se realiza de la misma forma que **beginDialog**. El único cambio respecto a esta, es que el outputLog presenta un mensaje de despedida frente a la conversación que se sostuvo con el usuario.

```
chatbot — swipl -S chatbot_19753546_GaeteLucero.pl — swipl — swipl -S chatbot_19753546_GaeteLucero.pl — 111...
?- endDialog(Chatbot, [{"[20/5/2018, 17:18]", "BeginDialog", "ID:", "0"}, [{"[20/5/2018, 17:18]", "Bot:", "Hola, mi nombre es Bot, y si quieres viajar, conmigo debes hablar. ¿Cómo debo llamarte?"}, [{"[20/5/2018, 18:11]", "Usuario:", "Gabriel"}, [{"[20/5/2018, 18:11]", "Bot:", "Gabriel ¿a qué capital regional deseas viajar? Puedes hacerlo a cualquier región de Chile. Yo te recomiendo el norte."}, [{"[20/5/2018, 18:17]", "Usuario:", "tengo ganas de viajar a Valparaíso"}, [{"[20/5/2018, 18:17]", "Bot:", "Valparaíso es ideal en esta época del año, no te arrepentirás. Viajar hacia allá cuesta $6.500 pesos ¿Desea confirmar su destino?"}], 40, OutputLog).
Chatbot = [{"[20/5/2018, 17:18]", "BeginDialog", "ID:", "0"}, [{"[20/5/2018, 17:18]", "Bot:", "Hola, mi nombre es Bot, y si quieres viajar, conmigo debes hablar. ¿Cómo debo llamarte?"}, [{"[20/5/2018, 18:11]", "Usuario:", "Gabriel"}, [{"[20/5/2018, 18:11]", "Bot:", "Gabriel ¿a qué capital regional deseas viajar? Puedes hacerlo a cualquier región de Chile. Yo te recomiendo el norte."}, [{"[20/5/2018, 18:17]", "Usuario:", "tengo ganas de viajar a Valparaíso"}, [{"[20/5/2018, 18:17]", "Bot:", "Valparaíso es ideal en esta época del año, no te arrepentirás. Viajar hacia allá cuesta $6.500 pesos ¿Desea confirmar su destino?"}, [{"[20/5/2018, 18:24]", "Bot:", "Hasta la próxima, espero haberte ayudado."}], [{"[20/5/2018, 18:24]", "EndDialog", "ID:", "0"}], 40, OutputLog).
?-
```

Figura 7 Uso de endDialog

3.4 logToStr(Log, StrRep)

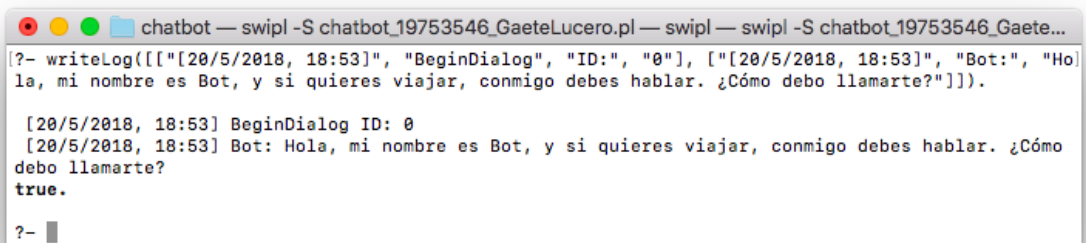
Este predicado permite expresar el Log de una conversación en forma de un string que mantenga un formato que sea más fácil de entender. La siguiente figura, muestra un ejemplo de uso, dado un OutputLog entregado por el predicado beginDialog.



```
chatbot — swipl -S chatbot_19753546_GaeteLucero.pl — swipl — swipl -S chatbot_19753546_Gaete...
[?- logToStr(["[20/5/2018, 18:53]", "BeginDialog", "ID:", "0"], ["[20/5/2018, 18:53]", "Bot:", "Ho]
la, mi nombre es Bot, y si quieres viajar, conmigo debes hablar. ¿Cómo debo llamarte?"]), StrRep).
StrRep = "\n [20/5/2018, 18:53] BeginDialog ID: 0\n [20/5/2018, 18:53] Bot: Hola, mi nombre es Bot
, y si quieres viajar, conmigo debes hablar. ¿Cómo debo llamarte?".
?- ]
```

Figura 8 Ejemplo predicado *rate*

Nótese que cada “\n” separa los mensajes. Esto se hace para que, en caso de que se estime ocupar el predicado **write**, el log adopte un formato aún mejor. Para comodidad del usuario, se ha añadido un predicado que no está dentro de los requeridos, el cual realiza esto. La siguiente figura muestra su utilización.



```
chatbot — swipl -S chatbot_19753546_GaeteLucero.pl — swipl — swipl -S chatbot_19753546_Gaete...
[?- writeLog(["[20/5/2018, 18:53]", "BeginDialog", "ID:", "0"], ["[20/5/2018, 18:53]", "Bot:", "Ho]
la, mi nombre es Bot, y si quieres viajar, conmigo debes hablar. ¿Cómo debo llamarte?"])).

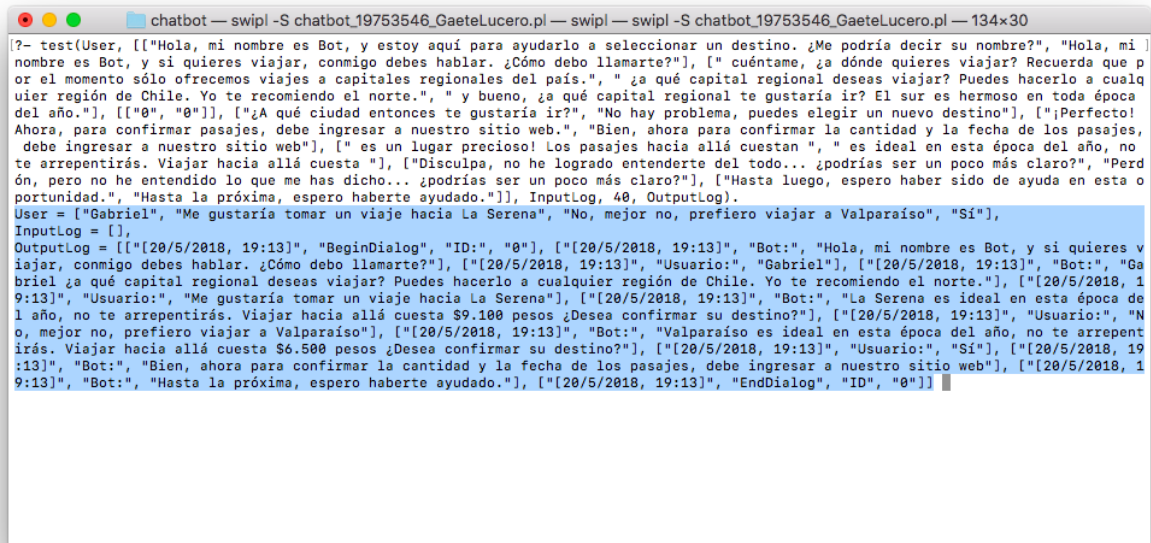
[20/5/2018, 18:53] BeginDialog ID: 0
[20/5/2018, 18:53] Bot: Hola, mi nombre es Bot, y si quieres viajar, conmigo debes hablar. ¿Cómo
debo llamarte?
true.
?- ]
```

Figura 9 Predicado *writeLog*

Se vuelve a mencionar, el predicado writeLog(Log) de la Figura 9, no está dentro de los requisitos, sólo se añade para explicar de mejor forma el formato entregado por el predicado **logToStr**.

3.5 test(User, Chatbot, InputLog, Seed, OutputLog)

Esta funcionalidad permite ilustrar/simular el desarrollo de una conversación entre un usuario “user” y un chatbot. El argumento user corresponde a una lista de strings, en las que en cada lista se encuentra el mínimo de mensajes para que la conversación tenga un inicio y un término adecuado. Para probar esta función, se brindan tres ejemplos de user dentro de los hechos del código fuente (en archivo **hechos.pl**), con el nombre user1, user2 y user3.



```
?- test(User, [{"Hola, mi nombre es Bot, y estoy aquí para ayudarlo a seleccionar un destino. ¿Me podría decir su nombre?", "Hola, mi nombre es Bot, y si quieres viajar, conmigo debes hablar. ¿Cómo debo llamarte?"}, {"cuéntame, ¿a dónde quieres viajar? Recuerda que por el momento sólo ofrecemos viajes a capitales regionales del país."}, {"¿a qué capital regional deseas viajar? Puedes hacerlo a cualquier región de Chile. Yo te recomiendo el norte."}, {"y bueno, ¿a qué capital regional te gustaría ir? El sur es hermoso en toda época del año."}, {"0", "0"}], [{"A qué ciudad entonces te gustaría ir?"}, {"No hay problema, puedes elegir un nuevo destino"}, {"¡Perfecto! Ahora, para confirmar pasajes, debe ingresar a nuestro sitio web."}, {"Bien, ahora para confirmar la cantidad y la fecha de los pasajes, debe ingresar a nuestro sitio web"}], [{"es un lugar precioso! Los pasajes hacia allá cuestan "}, {"es ideal en esta época del año, no te arrepentirás. Viajar hacia allá cuesta "}, {"Disculpa, no he logrado entenderte del todo... ¿podrías ser un poco más claro?"}, {"Hasta luego, espero haber sido de ayuda en esta oportunidad."}, {"Hasta la próxima, espero haberte ayudado."}], InputLog, 40, OutputLog).
User = ["Gabriel", "Me gustaría tomar un viaje hacia La Serena", "No, mejor no, prefiero viajar a Valparaíso", "Sí"],
InputLog = [],
OutputLog = [{"[20/5/2018, 19:13]", "BeginDialog", "ID:", "0"}, {"[20/5/2018, 19:13]", "Bot:", "Hola, mi nombre es Bot, y si quieres viajar, conmigo debes hablar. ¿Cómo debo llamarte?"}, {"[20/5/2018, 19:13]", "Usuario:", "Gabriel"}, {"[20/5/2018, 19:13]", "Bot:", "Gabriel ¿a qué capital regional deseas viajar? Puedes hacerlo a cualquier región de Chile. Yo te recomiendo el norte."}, {"[20/5/2018, 19:13]", "Usuario:", "Me gustaría tomar un viaje hacia La Serena"}, {"[20/5/2018, 19:13]", "Bot:", "La Serena es ideal en esta época del año, no te arrepentirás. Viajar hacia allá cuesta $9.100 pesos ¿Desea confirmar su destino?"}, {"[20/5/2018, 19:13]", "Usuario:", "No, mejor no, prefiero viajar a Valparaíso"}, {"[20/5/2018, 19:13]", "Bot:", "Valparaíso es ideal en esta época del año, no te arrepentirás. Viajar hacia allá cuesta $6.500 pesos ¿Desea confirmar su destino?"}, {"[20/5/2018, 19:13]", "Usuario:", "Sí"}, {"[20/5/2018, 19:13]", "Bot:", "Bien, ahora para confirmar la cantidad y la fecha de los pasajes, debe ingresar a nuestro sitio web"}, {"[20/5/2018, 19:13]", "Bot:", "Hasta la próxima, espero haberte ayudado."}, {"[20/5/2018, 19:13]", "EndDialog", "ID", "0"}]
```

Figura 10 Ejemplo de conversación simulada mediante predicado test

Nótese que la sección remarcada en azul, corresponde a la salida para el primer usuario definido (user1). Sin embargo, el programa queda esperando otra posible respuesta (los demás usuarios), por lo que al ingresar un ; (punto y coma), se muestran los resultados para el user2, para posteriormente mostrar los resultados de user3, y terminar con un user vacío, es decir, un usuario que no ingresó texto.

Cabe destacar que se puede ingresar cualquier otra conversación a User. Por otra parte, en la Figura 10, se ha ingresado la estructura del Chatbot, sin embargo, como se dijo anteriormente, no es necesario que el usuario la especifique, sino que el programa la puede determinar internamente de ser necesario.

La siguiente figura, muestra el resultado para el user2.

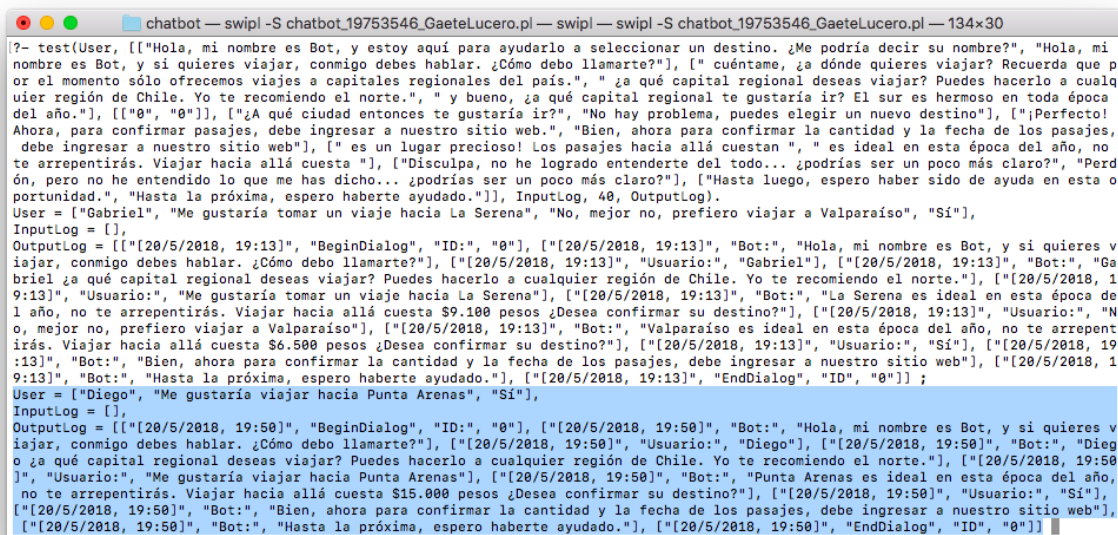


Figura 11 Muestra de múltiples usuarios funcionalidad test

CAPÍTULO 4. EN CASO DE FALLOS

En caso de errores fatales, se recomienda cerrar el programa swipl (abortar o terminar ejecución), y reiniciar siguiendo las instrucciones dadas al inicio de este manual.

Por otra parte, en caso de desplegarse un error de sintaxis, por favor lea las instrucciones que muestra el mismo intérprete y actúe acorde. Por ejemplo, en la Figura 12 se muestra un caso en el que no se respeta el formato de la funcionalidad **logToStr**, y es necesario escribir un término.

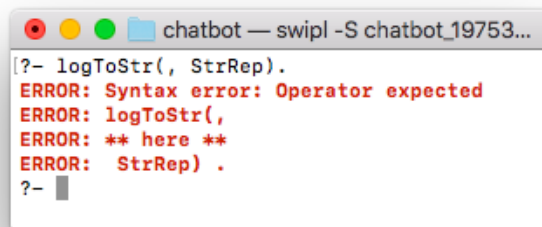


Figura 12 Error de sintaxis

Por otro lado, para los casos particulares de cada una de las funcionalidades, ver la sección 3 en este mismo manual correspondiente a cada funcionalidad.