Web-Dialogue Management for Insurance Products: IP.com

Süleyman Taner
Practis BV, The Netherlands and
Delft University of Technology, The Netherlands

Henk Koppelaar
Delft University of Technology, The Netherlands

ABSTRACT
The IP.COM concept enables an end-user to compose automatically an insurance product after conducting a dialogue with a knowledge-based system. This reduces the dependencies of insurance companies on both the IT and the actuarial expert. The system is able to adjust the dialogue interactively according to the specific needs of the users and asks for the relevant data needed for the desired product.

INTRODUCTION
The internal structure of insurances makes them exceptionally suited for automatic composition/configuration. Insurances consist of a number of common basic parts to configure. These parts will be referred to as the ‘building blocks’ of insurances. Different combinations of these building blocks lead to different insurance products.
From a man-machine dialogue, it should be possible to automatically compose insurance products out of a set of its building blocks. The ‘man’ is the client/user and the ‘machine’ is the automated web server of an insurance company. In this chapter we explain how we have built a program (via Kleene’s Algebraic machinery), called IP.Com, to learn and remember the dialogue and to configure from the user’s needs an insurance product on-the-fly. This work would have been impossible without the back-office deep reasoning automata by Willemse (1999).

BUILDING BLOCKS PRODUCTS

Regular expressions introduced by Kleene (1956) easily describe a dialogue between man and machine. From this formalization we obtain an abstract system to infer life insurance and policies composed from a limited amount of ‘building blocks.’ Let us use a simple introduction in the domain knowledge of the actuary field of mathematics, needed for life insurances. For example a life insurance policy in which, after a payment of one lump sum, the beneficiary receives an endowment if the insured is alive, can be reduced to three basic events. These three basic events belong to the set of building blocks common to all insurances: lump sum, endowment and alive.

From this wording we notice events and their administration. This is common in information systems design, but not explicitly used in dialogue systems. For instance Kleene created his algebraic formalism to describe the relation between an ‘agent’ and its environment. The agent detects events (Russell & Norvig, 1995) in its environment by observing properties of input stimuli it receives. In this section we will briefly explain our automation for life insurances based on Kleene’s formalism.

Remembering Dialogues

A program remembers a dialogue via pairs of current state and input to what the present output (symbol) and the next state must be. A transition graph visualises such. A node represents a state, an interconnecting arrow represents a transition to the next state. Each arrow is labelled with an input (symbol). Inputs trigger next state. A transition graph has a starting node and a terminal node. A graph is said to ‘remember’ a string of symbols if there exists a path through the graph such that the symbols found along the path match the symbols in the string. Dialogues can be learned from the user on-the-fly of a dialogue. This type of automated programming/construction of dialogues is treated in the sequel under the heading ‘Inversion: model inference.’

In order to enable a program to remember every occurrence we apply ‘Regular Expressions’ as devised by Kleene (1956) and according his fundamental theorem: every observable event (in a dialogue) is a regular event. Let $M$ be a
A Graphical User Interface (GUI) Testing Methodology
www.igi-global.com/article/graphical-user-interface-gui-testing/2643?camid=4v1a