Introduction

Introduction

Now, adding new features to my application is done by adding a few entries into a navigation list, (probably) writing some new rules and localizing some keys. That's it. For most of the stuff I get asked for, it takes more time to write the commit messages to my version control system than it does to actually implement the requested changes. The day I decide I revamp my interface I need to change about five templates and adjust one CSS file and I'm done. - Paraphrased from a post by Anjo Krank

What developer wouldn't want this kind of power at their fingertips?

What is DirectToWeb?

DirectToWeb describes a technology that STILL, some nine years after its introduction, points to a logical future for how data-driven web applications ought to be developed. DirectToWeb is an expert system applied to web applications. "The general architecture of an expert system involves two principal components: a problem dependent set of data declarations called the knowledge base or rule base, and a problem independent (although highly data structure dependent) program which is called the inference engine." http://en.wikipedia.org/wiki/Expert_systems The WebObjects developer that wishes to use DirectToWeb to create his application manipulates the rules in the rule base to result in the desired behaviour. The DirectToWeb frameworks contain the inference engine that is used to determine what rules will be fired.

D2W as Expert System

General information about Expert Systems below has been taken from http://en.wikipedia.org/wiki/Expert_systems and http://en.wikipedia.org/wiki/Inference_engine

An inference rule is a statement that has two parts, an if-clause and a then-clause. In D2W this is referred to as left hand side (LHS) and right hand side (RHS). The LHS is used to determine whether something is true and the RHS assigns a value if the LHS condition is true. When more than one rule is true, the more specific rule (i.e. the one with more terms on the LHS) will fire. In the case of the same specificity, rule priorities are compared.

One advantage of inference rules over traditional programming is that the rules use logic and reasoning that is similar to how humans reason. The disadvantage in the case of D2W is that traditional programmers find it unfamiliar and perhaps a little threatening to think like this!

When a conclusion is drawn from the rule system and a rule is executed, it is possible to retrace the reasoning that the system used to fire that rule (there is a special debug flag in Wonder that allows you to explicitly follow the rule system step by step as it works through the logic and makes choices about the application's behaviour).

The engine that drives and evaluates all these rules in D2W is called the Rule System. Given that you'll be writing the rules that are going to be evaluated by the rule engine it may be helpful for you to know what the system is doing behind the scenes. In general, the Rule System is an inference engine with three states. The first state "matches rules" that meet the current conditions of the application. The rules that match are passed to the second state called "select rules" which determines which rules will be executed (or "fired" in D2W terminology). D2W uses heuristics to determine which rules fire. In some cases the stock D2W rule system has limitations (for example it had limited significant keys used to evaluate rules, so the programmer needed to add additional keys so that rules would fire correctly) that Wonder (ERD2W) has addressed using a more modern "select rules" process that removes the original limitations. The selected rules are passed to the third state "execute rules". The rules that execute will change the context of the application, whether through updates in the database or presenting a different UI to the user. This will cause the rule system to cycle back to the first state, ready to match the appropriate rules for the new context of the application. For much greater detail about the mechanics of the D2W rule system please see The D2W Rule System.

As an expert system that has been designed to capture the process of developing a data driven web application, DirectToWeb inherits the advantages and disadvantages of any expert system: http://en.wikipedia.org/wiki/Expert_systems#Advantages_and_disadvantages

Advantages:

- Provides consistent answers for repetitive decisions, processes and tasks
  - The normal WO developer's tasks involve programming similarly functioning components that are created to perform common set of repetitive tasks on entities. D2W allows the developer to leverages the similarities and pulls out the common tasks (for example: query, list, inspect, edit, select, confirm) to put them in their own templates and apply them to your entities (defined by the EOModel).
- Holds and maintains significant levels of information
  - The D2W rules system knows the tasks that you will want to use in a typical application for a typical entity. It also knows when you need the behaviour for a given component to be different from the default because you have told it so.
- Encourages organizations to clarify the logic of their decision-making
  - Once the organization determines their business rules, the developer has a logical way to codify those rules into the rule base. D2W uses "rule models" that are edited using a specific tool for the job. Once the codified behaviour is observed in the application, the business logic can be verified by the organization and quickly changed if it is incorrect.
- Never "forgets" to ask a question, as a human mind
  - Once the rules are established, the rules are always considered to determine the "correct" behaviour for the given situation.

Disadvantages:

- Lacks common sense needed in some decision making
  - Not a severe shortcoming for WebObjects developer since the practical range of UI options for a finite set of data types is also finite
- Cannot make creative responses as human expert would in unusual circumstances
  - The development of a website doesn't often present unusual circumstances and when it does, traditional completely customized component-based WO development is always available as an option.
- Domain experts not always able to explain their logic and reasoning
  - This is not likely true of the process of developing WebObjects applications
- Errors may occur in the knowledge base, and lead to wrong decisions

    - This is true of any application! You have to test and correct errors.
- Cannot adapt to changing environments, unless knowledge base is changed
    - This is true of any WebObjects application, but based on Anjo's description at the top of this page, the scope of the work to change a D2W application is potentially much less.

Historical Context of D2W