DBPowder: RDBMS 活用フレームワーク
—RDBMS を身近に扱おう—

1. 背景

RDBMS は、データを管理・利用するための強力なコンポーネントであり、様々なアプリケーションのバックエンドとして広く利用されている。しかし、RDBMS の活用は容易ではない。容易でない背景として、以下(ア)～(エ)に示す4つが挙げられる。

適切なリレーショナルスキーマの構築は難しい(ア)。RDBMS の対象となるデータに対して適切なスキーマを提供するためには、リレーショナルモデル理論への十分な理解と、データの運用形態についての十分な分析が必要である。これらはいずれも容易ではない。

また、**スキーマ内のテーブル連携を手軽に実現できるアプリケーションに乏しい(イ)**。適切なスキーマを構築できたとしても、RDBMS をそれだけで利用するのは難しい。RDBMS を利用するためには、そのスキーマに特化した専用アプリケーションの構築が必要である。

ところが、上記の専用アプリケーション作成や、その他 RDBMS を用いたアプリケーション構築のための **RDBMS プログラミングは煩雑である(ウ)**。RDBMS プログラミングは、学習コスト及び作業工数は高く、現状ではスキルの高いプログラマ又はチームでの開発が不可欠な分野である。

RDBMS プログラミングが煩雑であることから、**開発後のスキーマの再変更は非常に困難である(エ)**。RDBMS プログラミングでは各コンポーネントが密に連携しあう。従って、スキーマ変更時に同時に変更せねばならない箇所が多くなり、全体を正しく変更するのはしばしば困難となる。開発現場において RDBMS のスキーマ再変更は、運用に供しているシステムのみならず、開発後期の段階でもタブー視されることが多い。

2. 目的

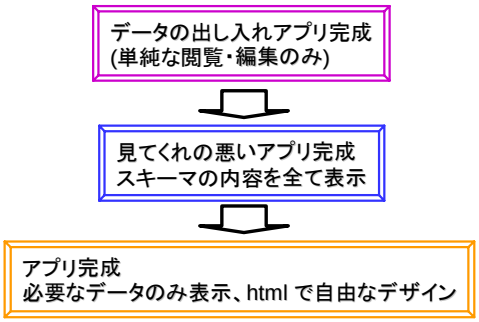
本プロジェクトでは、背景で述べた4つの問題点を解消するソフトウェアとして、DBPowder を開発した。つまり DBPowder は、以下のようなソフトウェアである (表 1)。

表 1: DBPowder

ア) 適切なリレーショナルスキーマの構築を助ける
イ) スキーマ内のテーブル連携を手軽に実現できるアプリケーションとして作用する
ウ) RDBMS プログラミングの煩雑さを軽減する
エ) 開発後のスキーマの再変更を可能にする

本プロジェクトの範囲では、最近特に利用頻度の高い、RDBMS にウェブを組み合わせた開発を主ターゲットとする。つまり、RDBMS を用いたウェブアプリケーションの構築を助けるフレームワークとして DBPowder を提供する。

1. DBPowder-schema
   - データ構造(スキーマ)を書く
2. DBPowder-navigation
   - ウェブページの動きを書く
3. DBPowder-html
   - ウェブページを書く



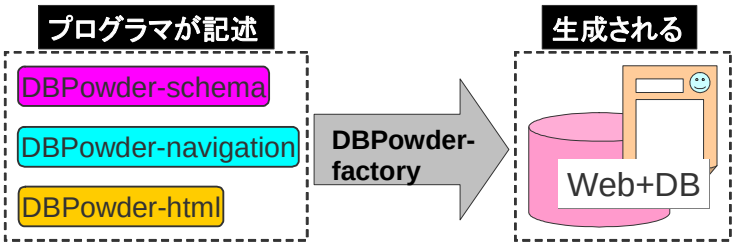


図 1 DBPowder-schema, navigation, html, コードジェネレータ(DBPowder-factory)

3. 開発の内容

DBPowder では、RDBMS を用いたウェブアプリケーションの構築を助ける言語 DBPowder-schema, DBPowder-navigation, DBPowder-html と、これらの言語に基づいたコードジェネレータ DBPowder-factory を提供する。それらの概要を図 1 に示す。

DBPowder-schema のみから RDBMS を用いたウェブアプリケーションを構築できる一方で、DBPowder-navigation, DBPowder-html と組み合わせることで本格的なウェブアプリケーション開発にも進めることを特徴とする。さらに、複雑なロジックや外部モジュールとの連携が必要な際には、DBPowder-API を用いてカスタムコードの記述が可能である。これらの機能は表 1 のア)ウ)エ)の実現に寄与する。

開発の出発点となる DBPowder-schema の例を図 2 に示す。上側には例を示し、下側にはそれに対応する E-R 図を示す。

DBPowder-schema は、スキーマ(データ構造)を記述するための言語である。各行はエンティティ(実体)またはアトリビュート(属性)を表す。各行は階層構造になっており、親要素が子要素を保持する関係にある。DBPowder-schema では、リレーショナルスキーマで必須となる主キーや外部キーの定義をプログラマが行う必要がないため、非常に簡潔な記述が可能となっている。同名エンティティの重複記述があった場合、それらはマージされる。これにより、直感的にわかりやすい階層構造をベースにしつつも RDBMS が扱うリレーショナルス

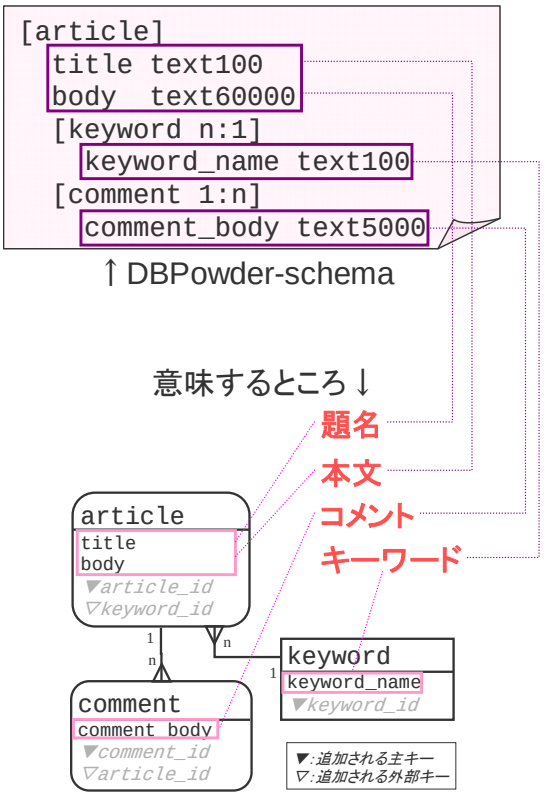


図 2 DBPowder-schema

キーマを記述するための十分な表現能力を持つ。

DBPowder-schema のみを与えて DBPowder-factory でコード生成すると、図 3 のようなウェブアプリケーションが得られる。このアプリケーションは、定義したスキーマにデータを入力・閲覧・検索・編集する機能を持つ。複数テーブルの連携をはじめから実現している点に特徴がある。また、このウェブアプリケーションは DBPowder-API から構成されているので、このウェブアプリケーションを改造して独自アプリケーションを構築することも可能である。

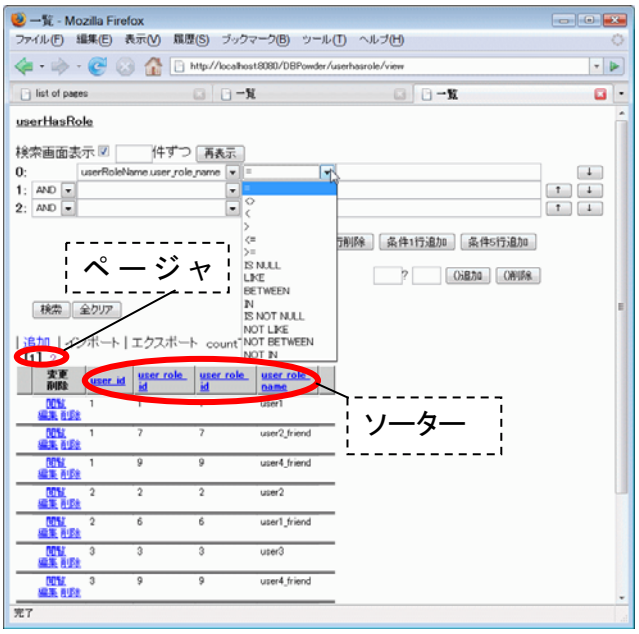


図 3 DBPowder-schema から生成されるアプリ

DBPowder-schema でスキーマを与えた後、DBPowder-navigation, DBPowder-html を記述することで独自アプリケーションを構築することもできる。DBPowder-schema, navigation, html の連携を図 4 に示す。DBPowder-navigation ではウェブページの遷移を記述し、DBPowder-html では各ページの構成を記述する。DBPowder-html を与える前後の様子を図 5 に示す。

4. 従来の技術(または機能)との相違

従来の類似技術は、主キー外部キーなどリレーショナルスキーマを意識する必要がある

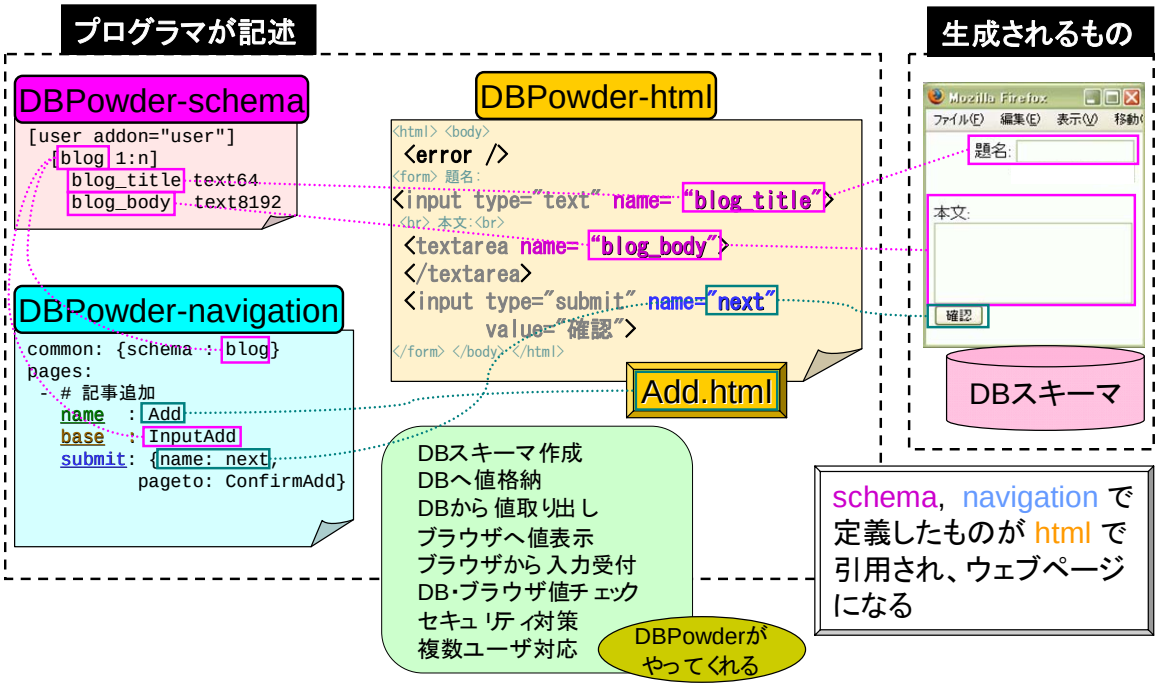


図 4 DBPowder-schema, navigation, html の連携

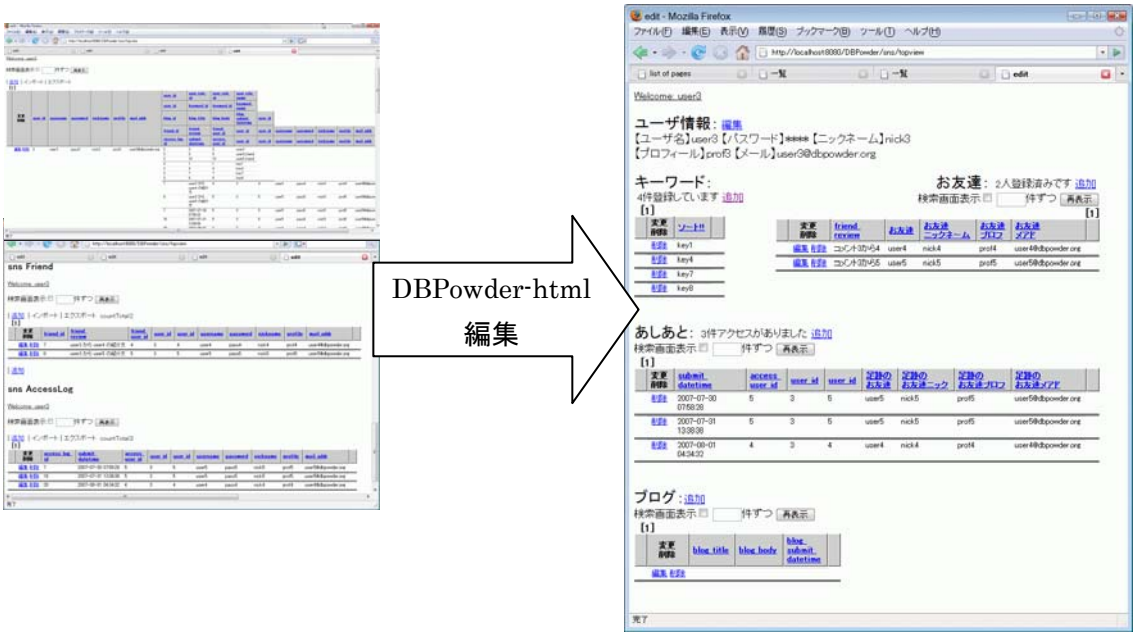


図 5 DBPowder-html 編集の前後比較

か、或いは GUI などで開発は容易だがカスタマイズの困難なものがほとんどである。

DBPowder の特徴は、表 1に示した本プロジェクトの目標を同時に満たしたことにある。スキーマに関する定義を DBPowder-schema に集約し、DBPowder-schema の簡潔さを追求することで、リレーショナルスキーマにおける開発の煩雑さと再変更の困難さを克服できた。DBPowder-schema では主キーや外部キーは自動的に補われるが、明示的な指定も可能であり、必要に応じてカスタマイズできる。

また、DBPowder-schema のみでウェブアプリケーションを構築できる一方、DBPowder-navigation, DBPowder-html と連携させることで、ページ遷移やレイアウトを様々に実現できる。プログラマは、開発の簡略さと柔軟さの両方を享受できる。

5. 期待される効果

従来 RDBMS をベースとしたアプリケーションの開発では、複数人の関与が必要とされることが多い。DBPowder を用いることで開発工数を大幅に下げることができ、場合によっては 1 人でも RDBMS をベースとしたサイトの構築・管理が可能になる。

6. 普及(または活用)の見通し

多くの開発・運用・保守現場では人員不足にあえいでおり、DBPowder の効果は大きいと考えている。例えば提案者(私)の日々運用業務は、DBPowder による開発工数削減なしには成り立たない。利用方法を整備して公開するべく、準備を進めている。

7. 開発者名(所属)

村上 直(高エネルギー加速器研究機構)

(参考)開発者URL

http://www.dbpowder.org/ (準備中)