共同研究報告

T-NEXT 活用支援システムの開発による 学生エンロールマネジメント改善

共同研究メンバー
○出原至道（多摩大学）、志賀敏宏（多摩大学）、中村有一（多摩大学）
（○代表、執筆者）

1．背景と目的

多摩大学で学籍管理に使用されている T-NEXT システムは、学生の情報をひとりひとり見ていかないと履修や単位の状況が確認できないうえ、ホームゼミの履修状況や卒業に必要な単位の照会といった重要な情報が画面の下の方にスクロールしないとみられない。このような課題に対して、T-NEXT の定期的なシステム改善で対応できる可能性があるとはいえ、細かい表示ルールは、大学の教育システムが変更されたり、使い込んでいく過程で気づいたりして次々に出てくるもので、公式なバージョンアップを待っていては小回りがきかない。このため、ユーザが自由に設計可能な支援システムを試作する[1]。

2．手　法

2.1　条　件

本システムは、T-NEXT システムの利便性を高めるにあたって、T-NEXT 本体には一切手を加えないことを第一条件とする。この前提で、本システムの実装手法を検討する。

T-NEXT では、ページの拡張子が JSP であることから、Java Server Page が利用されている。この場合、スクリプトはサーバサイドで実行され、クライアントユーザ側にはスクリプトは公開されない。したがって、データベースへのアクセスについて、クライアント側で内部データベースに直接アクセスすることは許されていない。T-NEXT 本体に手を加えないという条件のもとでは、データへのアクセスは、提供されているウェブインタフェースを通して行う必要がある。

T-NEXT のウェブインタフェースのリクエストの送信は、一般的な HTML の POST メソッドを通して行われている。本システムでは、本来人間が操作して発行される一連の POST メソッドの送信をシステムが代行することにより、人間の操作負担を軽減する。

[1] この仕組みは、ユーザサイドでの活用支援システムの実験的な検証を目的としている。実用とするには、規則などの整備が必要であると考えられる。

（原稿受理日　2014.10.26）

2.2 実装手法の検討（1）：独自クライアント

ウェブインタフェースについて、独自に T-NEXT へのアクセスを行うための独自のクライアントアプリケーションを実装し操作を改善することは、抜本的にユーザインタフェースを設計することができるため、自由度の高い実装が可能である。たとえば、複数の条件を設定しておき、そのいずれの条件に当てはまっても一覧でアラートを出すようなアプリケーションが開発可能である。

しかし、独自クライアントを実装する場合、次の点が問題となる。

1. T-NEXT のように安全性に配慮した認証機構をそなえたウェブインタフェースを持つシステムに対応して、その機構までふくめて独自のクライアントを作成することは、作業負担が大きい一方で、セキュリティリスクが高まる。
2. クライアントを独自実装する場合、オペレーティングシステムに依存する。現在、正式なサポートの有無について問わなければ、Windows, MacOS, Linux, 各種スマートフォンからシステムが利用可能であるが、この利便性が損なわれる。

   このため、今回は、独自クライアントを実装するという案は採用しない。

2.3 実装手法の検討（2）：ユーザスクリプト

ユーザスクリプトとは、ブラウザ上で画面が表示される際にクライアント側で実行されるプログラムである（図 2）。通常の JavaScript がサーバ側で設計されクライアントに送り込まれてクライアント上で実行されるプログラムであるのに対し、ユーザスクリプトは、サーバ側から独立してクライアント側で設計・動作する。このため、サーバ側から見た場合、ユーザスクリプトは存在が認識されず、クライアントからのアクセスは、通常のアクセスと判別できない。

アクセス認証やセキュリティの確保はブラウザが担当するため、実装者の作業負担は軽く、セキュリティリスクも基本的には従来のブラウザによるアクセスと変わらない[2]。また、ユーザスクリプトはオペレーティングシステムではなくブラウザに依存する。一般にブラウザの方が汎用性が高いため、ユーザスクリプトはさまざま機材に対応させることが用意である。

一方で、画面や機能の設計は、独自クライアントを作成する場合に比べて自由度が低い。今回は、基本的な画面の設計や操作感を保ったまま活用支援を行うことを目的とするため、この点は問題とならない。

したがって、今回はユーザスクリプトによる実装を採用する。

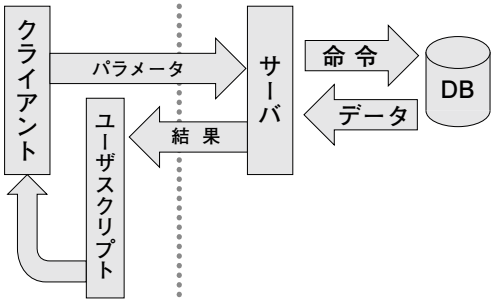

図 2　ユーザースクリプトの概要

ユーザスクリプトの実行環境としては、Mozilla Firefox 上で動作する Grease Monkey プラグインが主流であった。これに対し、Google Chrome は、特別なプラグインなしで同様のユーザスクリプトを実行可能としている。この双方に対応したスクリプトを作成することで、Firefox または Chrome

[2] ユーザスクリプトがサーバ側のスクリプトを直接実行するなどの場合は、サーバ側からクライントを操作される危険性が高まる。これは、今回の場合当てはまらない。

のいずれかが動けば利用できる仕組みとなる。これには、Windows, MacOS, Linux, Android スマートフォン，Windows Surface が含まれる。

3．実　装

3.1　学生リストからの情報の取得

今回の実装では、画面表示のために送られてきた学生リストについて、人間が順番にそのリンクをクリックする動作をシステムに代行させ、その結果の中からルールベースで必要な情報を取り出す。このため、現在利用中のシステムと同様に、学生リストを何らかの方法で作成する必要がある。今回は、事務局で作成される担当学生リストを利用した。これは、各教員が学生リストを表示したときの初期設定である。

学生に関する情報は、本来の表示をできるだけ変更しないで表示する。今回の実装では、本来表示される画面上の学籍番号と学生名の文字を縮小し、その右に表示のためのプレースホルダを設け、ここにデータを流し込むことで情報を提示した（図 3）。AJAX 技術を使用して、学生情報への内部的なネットワークアクセスは非同期で行われ、結果は順次表示画面に反映される（図 4）。サーバ側の処理速度の限界から、それぞれの学生情報のアクセス間隔を、実験的に 500［ms］と決定した。これより頻繁なアクセスでは、サーバからのレスポンスが安定しなかった。

非同期読み込みを利用することにより、通常の画面表示時間は、本システムを利用しても変化せず、約 0.6 秒であった。一方、30 名のリスト表示が完了するまでには、およそ 85 秒必要であった。

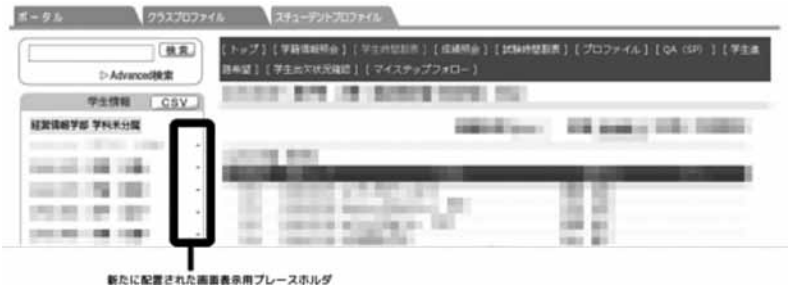

図 3　情報表示用プレースホルダの設置

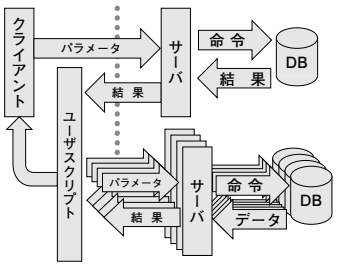

図 4　個別学生情報の取得

3.2　実装例 1：情報の追加表示

実装例として、総取得単位数の一覧表示を行った（図 5）。

取得単位数の画面で学生リストを表示する際に実行されるユーザスクリプトで、順に、それぞれの学生の取得単位数ページにアクセスし、その中に現れる総取得単位数の数値を取得して、プレースホルダに出力した。

これによって、取得単位数が著しく低い学生を簡単に発見することができる。

3.3　実装例 2：ルールによるアラート表示

実装例として、ルールによる学生状況のアラート表示を行った（図 6）。

登録科目一覧の画面で学生リストを表示する際に実行されるユーザスクリプトで、順に、それぞれの学生の登録科目一覧ページにアクセスし、登録科目に「ホームゼミナール」の文字列が出現するか判定し結果を「◎」「－」で、プレースホルダに出力した。

これによって、担当学生がホームゼミナールの履修漏れ学生を簡単に発見することができる。プレゼミナールの担当学生には当然「－」が表示されている。この他に 1 名の学生が今期のホームゼミナールを登録していないことが検出されている。現在の判定条件では、他の教員のホームゼミナールを登録してしまっているミスは検出できない。

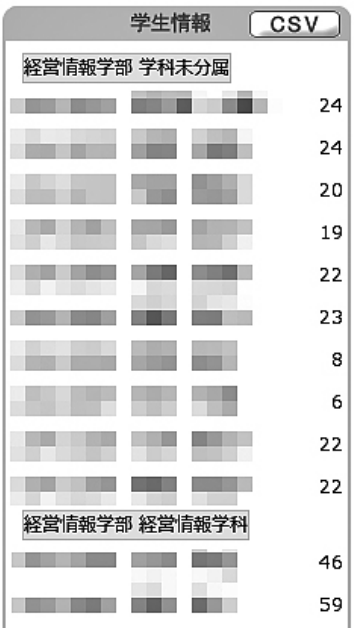

図 5　取得単位数の表示

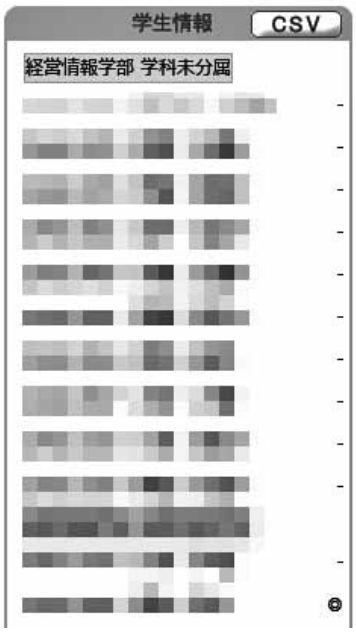

図 6　ホームゼミナール登録の判定

4．課　題

現在の T-NEXT システムでは、学生リストをユーザが自由に作成することはできず、あらかじめ事務局で設定されたリストを利用するか、検索によって学生を列挙する必要がある。この点を自動化することができれば、手元で作成した学生グループを一覧することが可能になり、複数のグループの管理を行うことが可能になる。しかし、この点はトラフィックが増大することが予想される点から、実装を見送った。T-NEXT のバージョンアップで対応されることを期待したい。

判定条件や表示項目については、それぞれの教員で必要とする内容が異なっている。これらは全てユーザスクリプト内に記述する必要がある。これを、ルール記述言語と、そのルールを解釈して動作するスクリプトとに分割することができれば汎用性が高まる。しかし一方で、必要とされるルールが教員間で著しく異なっていることは想定しにくいことと、一方で汎用ルール記述言語を定義し実装することが困難であることから、個別にルールを実装することで対応し、ルール記述言語の実装は行わなかった。