論　説

小テスト記述言語の定義と実装[1]

小川　浩[2]

Simple Quiz Description Language, design and implementation

Hiroshi Ogawa
Kanagawa University

【要約】 Instructional Design（ID）を用いた教育プロセスにおいては、学習効果を計測するために授業前後および授業中に実施するテストが重要であり、教材を設計する前にゴールを設定するためにテストを作成する。大人数授業において個別学生の学習状態を把握するためには、e-learning システムを用いたテストの活用が ID の観点からは効果的と予想される。しかしながら、神奈川大学における e-learning の利用状況は資料配付とレポート回収に集中しており、テスト実施のプラットフォームとしての利用は限定的である。

本稿では、e-learning で頻回のテストを実施する際の障害となっているテスト作成のコストを、単純なテスト記述言語を用いて LMS でインポートできる問題セットを生成することで下げることを試みた。同様の試みは多く行われているが、多くの場合は作成ツール事態が複雑すぎるあるいは対応できる問題種別が少なすぎるなど単純すぎて、実際の作問には使いにくい。試作したシステムでの作問所要時間は dotCampus の内蔵問題エディタでの作問時間より十分短く、大量にテストを作成する際の優位性が示された。

【キーワード】 Instructional design　e-learning　QUIZ

目　次

1　本研究は2017年度経済貿易研究所共同研究調査助成金の援助を受けて実施した。
2　神奈川大学　経済学部　santa@econ.kanagawa-u.ac.jp

1．Instructional Design とテスト

1.1. Instructional Design におけるテストの役割

教育プロセス全体を、学習を成立させるためのシステムとみなし、システムがより効率的に、そして目標となる学修効果を達成するように改善する方法の一つとして、インストラクショナルデザイン（Instructional Design、以下 ID と略記）と呼ばれるアプローチがある。［ディック、ケアリー、ケアリー 2004］によると、このアプローチによる教育システムは表 1 の10ステップから構成されている。

表 1 で 5 番目にきている評価基準の開発とは、端的に言えば到達度を測定するためのテスト作成である。つまり、ID による教育プロセス設計では、教え方の選択（6 の教授方略の開発）や教材開発（7 の教材の開発と選択）より前に「意図した学修が達成できたとはどういう状態か」を定義し、その状態を計測するためにテストを作成することになる。この考え方を、［鈴木 2002］では「テストで出入口を明確化するのは、教材で誰に何を教えようとしているかをはっきりさせることなんだ」と表現している。表 2 に ID で用いられるテストとその目的について整理しておく。

このように、ID においてはテストが非常に重要な意味を持っている。この文脈でのテストは、「点数を付けて順序づけするためのテスト」ではなく、学生の学修状態を確認し、適切なフィードバックを与えるための計測手段のことである。そのため、テストの実施回数は 1 回には限らないし、テストの目的次第では教材を確認しながら受験しても問題はない。

表 1　ID での教育システムの構成

1	ゴールを識別するためのニーズアセスメント
2	教育分析（Instructional Analysis）の実施
3	学習者分析とコンテキスト分析（Context Analysis）
4	パフォーマンス目標の作成
5	評価基準（Assessment Instruments）の開発
6	教授方略の開発
7	教材の開発と選択
8	形成的評価（Formative Evaluation）の設計と実施
9	インストラクションの改訂
10	総括的評価（Summative Evaluation）の設計と実施

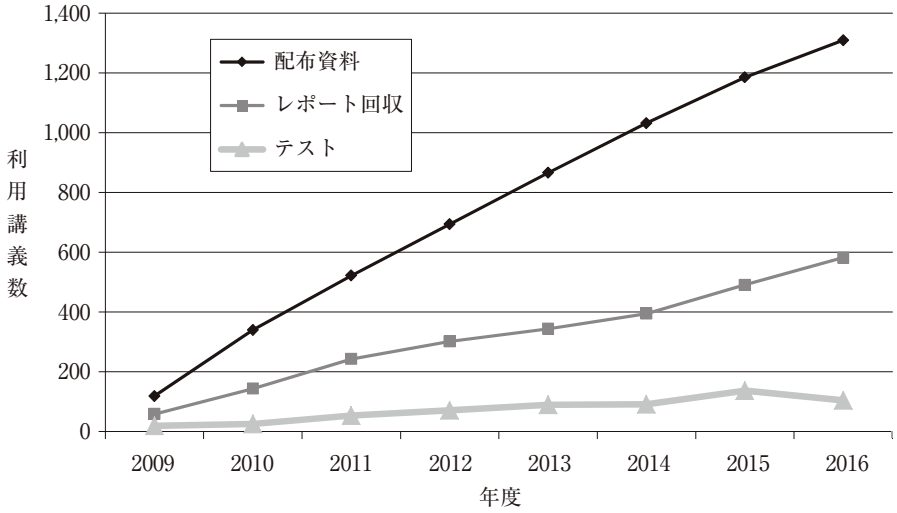
出所：［ディック、ケアリー、ケアリー 2004］第 1 章より筆者作成

表 2　ID で用いられるテストの種類と目的

テストの種類	実施タイミング	目　　的
前提行動テスト	授業前	当該授業で必要な前提知識の確認（必要に応じて事前学習を追加）
プリテスト	授業前	授業で扱う内容を授業前にどの程度理解しているかの確認
模擬テスト	授業中	学修中のフィードバック
ポストテスト	授業後	授業で扱った内容の理解を確認する。理解が不十分な部分のインストラクションを改善する。

出所：［ディック、ケアリー、ケアリー 2004］第 7 章より筆者作成

図 1　dotCampus の機能別利用講義数推移



資料：「2016年度指示種別利用講義数」神奈川大学メディア教育支援室

1.2. 神奈川大学における e-learning の利用状況

上述のように ID の考え方では授業前と（場合によっては授業中にも）テストを頻回で実施する。そのため LMS を用いたテストの自動化はインストラクターの負担を減らすための有効な方法である。

神奈川大学では2009年度から全学で利用可能な LMS として dotCampus[3]を導入している。dotCampus には、資料配付、レポート回収、アンケート、テスト、SCORM 教材の実行などの機能があるが、講義で実際に利用されている機能は圧倒的に資料配付、次いでレポート回収であり、ID の考え方では極めて重要であるテスト機能の利用は伸び悩んでいる（図 1 ）。さらに利用している講義の中での利用割合ではテストの占める割合はむしろ時系列的に低下してきており、利用講義が拡大したといっても、ごく初期から使っていたユーザー以外は紙で行っていた作業をシステム上に移しつつあるに過ぎない状況であることが分かる。

この点について筆者が同僚に行ったヒアリングでは、「テストの使い方が分からない」「作問が面倒で負担が重い」などのシステム対応で解決可能である意見もあり、LMS における作問を簡単に行う方法が求められている可能性は高いと考える。

以下では、教育改善のためにテストを簡単に、大量に作成することを前提としたテスト作成支援言語を提案し、その実装例について紹介する。

2．小テスト作成システムに望ましい要素と既存の作成システムサーベイ

2.1. 小テスト作成システムに必要なこと

上述のように、LMS を用いて小テストを頻回に実施する際に問題となる点は「作問に手間が

3　http://dotcampus.jp/

かかる」である可能性は高い。ここで、小テストを作成するシステムにとって望ましい要素を確認しておこう。

2.1.1. 作問が簡単

テストを多用するIDに基づく授業を実施するためには、作問が簡単であることは極めて重要である。また、多くの教員は過去に作成したテストをテキストとして保存していることを考えれば、テキストベースで作成して、システムに流し込むことで一括して作問できることが望ましい。

2.1.2. 表現力が豊富（数式や図の利用）

数式や図をどの程度テストの中に組み込めるかはLMSの性能に依存するが、全く使えないということは通常ないと予想できる。図は最低でも1つ、数式については問題文の任意の位置で使えることが望ましい。

2.1.3. 作った問題が特定LMS以外でも利用可能（vendor lock-inの回避）

LMSの乗り換えは別に珍しい話ではなく、教育プロセスで必要な機能が充実しているシステムに変更することはむしろ当然である。ただし、テスト機能はLMSによって異なることが多いため、特定LMSでしか使えないデータ形式で作成してしまうと他のLMSでは使えない、いわゆるベンダーロックイン（vendor lock-in）が発生してしまう。

この問題を回避するためには、多くのLMSが持っている機能に限定したテストを中間的なフォーマットで作成しておき、LMSにインポートする際に形式変更するような処理フローとすることが考えられる。

2.1.4. LMSだけではなく、紙ベースでの試験にも流用可能

LMSで実施するテストは参照や学生間の相談をコントロールすることが実質的にはできない。そのため、成績評価に必要なテストとして実施する際には紙ベースで教室試験を実施する必要が出てくることが多い。この際、LMS用に作成したテスト問題をそのまま紙ベースでの試験作成に流用できると便利である。

2.2. 既存小テスト作成システムのサーベイ

LMS向けの小テスト作成システムは既に多数存在している。ここでは、フリーソフトウェアとして配付されており、多くの利用者が改良を加え続けているMoodle[4]を対象とした小テスト作成システムとdotCampusの内蔵問題エディタについてサーベイを行う。

2.2.1. 作問ツールの概要

Moodle2Word

Wordの機能拡張として実装されている。Moodleからの Exportも Importもサポートしており極めて高機能。元々がWordなので、紙のテストとしての出力にも適している。ただし、高機能であるため使い方が難しい。本来はmoodle bookと呼ばれる本形式のコース全体を作成するツールであり、作問機能は全体のごく一部。教科書を作るツール。

Hot Potatoes

Webでの問題作成一般のツール。Moodle専用というわけではなく、Moodle側に専用のプラ

4 http://www.moodle.org/

表 3　既存のテスト作成方法（Moodle および dotCampus）

ツール名	簡単	表現力	Lock-in 回避	紙への流用
Moodle2Word[5]	×	○	×	○
Hot Potatoes[6]	△	○	○	×
QuEdit[7]	△	○	×	×
e 問つく朗[8]	△	○	×	×
Word の蛍光ペン[9 10 11]	○	△	×	○
Missing Word 形式[12]	○	×	×	△
Aiken 形式[13]	○	×	○	○
GIFT 形式[14]	×	○	×	×
dotCampus 内蔵エディタ	△	○	×	×

グインをインストールすることで使えるようになる。問題作成は対話的な GUI で行うようになっており、誤答に対するレスポンス設定なども含め、非常に細かい設定まで可能。

QuEdit

Moodle の GIFT 形式のファイルを出力するスタンドアロンの GUI 問題作成ツール。GIFT の読み込みも可能。GIFT は Moodle のネイティブファイルフォーマットなので、Moodle が扱える問題は作成できる。ただし誤答に対するレスポンス設定などの機能は無い。

e 問つく朗

Moodle の拡張モジュールとして実装されており、標準の問題エディタよりは人間の入力が少ない GUI 的作問が可能になっている。選択問題については CSV 形式で作成したものをインポートする機能がついている。

Word の蛍光ペン

Moodle の穴埋め問題作問は、複雑な制御文字列を埋め込む必要があるため穴埋め問題作成に特化したツールが数多く作成されている。特に Word の蛍光ペン機能を用いて、蛍光ペンでマークした部分を穴埋め問題として GIFT 形式で出力するツールは複数の人、グループが作成している。元が Word ファイルであるので、紙に出力してテストとして利用することも可能な実装が多い。

dotCampus の内蔵エディタ

当然のことながら dotCampus の全機能を利用可能。GUI で作問できるが、デフォルトで設定

5　http://www.moodle2word.net/
6　http://hotpot.uvic.ca/
7　https://www 1 .hus.ac.jp/~fukai/
8　http://ns 1 .shudo-u.ac.jp/~nakanisi/paper/Remedial2015a.pdf
9　http://www.jsise.org/taikai/2016/program/contents/pdf/C 3 - 3 .pdf
10　http://klibredb.lib.kanagawa-u.ac.jp/dspace/bitstream/10487/10344/ 1 /47%282%29- 3 .pdf
11　http://www.itc.u-toyama.ac.jp/moodle 3 /tool/#Jtest
12　https://docs.moodle.org/34/en/Missing_word_question_format
13　https://docs.moodle.org/34/en/Aiken_format
14　https://docs.moodle.org/34/en/GIFT_format

図２　GIFT フォーマットでの作問例[15]

```
What's the answer to this multiple-choice question? {
  ~wrong answer#feedback comment on the wrong answer
  ~another wrong answer#feedback comment on this wrong answer
  =right answer#Very good!
}

//From The Hitchhiker's Guide to the Galaxy
Deep Thought said " {
  =forty two#Correct according to The Hitchhiker's Guide to the Gal-
axy!
  =42#Correct, as told to Loonquawl and Phouchg
  =forty-two#Correct!
}  is the Ultimate Answer to the Ultimate Question of Life, The Uni-
verse, and Everything."

   42 is the Absolute Answer to everything.{
FALSE#42is the Ultimate Answer.#You gave the right answer.}
```

すれば変える必要が無いようなパラメータまで１問ごとに入力する必要がある上、どのパラメータがどのように学習者に表示される問題に影響するのかが分かりづらいため初心者が作問する際は試行錯誤が必要となる。今回実装したシステムとの比較は後述する。

2.2.2.　ファイルフォーマットの概要

Moodle XML

Moodle の全機能を記述できる XML ファイル。XML なので直接人間が書くことは期待されていない。完全に機械むけのファイルフォーマット。

GIFT フォーマット（図２）

Moodle が持つ各種作問機能を充分に活用できる高機能フォーマット。テキストファイルの中に制御構造を埋め込んだ形式になっており、プログラマならともかく一般ユーザに直接書かせることはかなり困難。上述の問題作成支援ツールの多くは、GIFT フォーマットで問題ファイルを出力する。

Missing Word フォーマット

単純なテキストファイルで、比較的簡単に穴埋め形式の作問が可能になるフォーマット。ただし、穴埋め問題しか作成できない。

Aiken フォーマット（図３）

単純なテキストファイルで単一選択問題を作成できる。正解選択肢が１つしか設定できない、

15　https://docs.moodle.org/26/en/GIFT_format#Feedback　より引用

図 3　Aiken フォーマットでの作問例[16]

```
Which LMS has the most quiz import formats?
A) Moodle
B) ATutor
C) Claroline
D) Blackboard
E) WebCT
F) Ilias
ANSWER: A
```

穴埋め問題が作成できない、などの機能的な問題はあるが、紙にそのまま書いた形式とほぼ同じ形式で作問できるため、既存の資源の流用、あるいは紙に印刷しての再利用などはやりやすい。

ただ、図版などを組み込む機能はついていないため、表現力という点では問題が残る。

2.3. 既存システムの問題点と改善可能な点

既存のシステムの概要は表 3および上述の通りであるが、目標とする機能レベルで大きく 3つに分類可能である。

1．問題作成の全機能が使えることを目指す
2．特定機能に限定して作問をサポートする
3．機能をできる限り限定して、楽に作問できるようにする

以下に、本論文での評価を示しておく。ただし、評価の前提となるのは図 1 に示すようなこれから LMS を利用したテストを導入する利用者段階にある組織である。既にテストを利用している講義が多く、学習者の反応についてのデータ蓄積、分析が進んでいるようなケースを想定しているものではない点に注意されたい。

2.3.1. 全機能サポートシステムは不要

LMS の全機能をサポートしようとした場合、使い方が非常に複雑になりがちである。特に誤答分析を含んだフィードバック系の記述は、もともとどのようにフィードバックを設定するかの意思決定にも関わるためハードルが高い。評価を含むフィードバックは ID のプロセスではテスト作成より後の部分（表 1では 8 番目）であり、テスト作成に必ず付随するものではない。e-learning を単体で行う場合はともかく、教室でのインストラクションとセットで行う場合には、フィードバックについてはインストラクターが行うという選択も可能である。

もちろん、迅速なフィードバックはよりよい理解をもたらす可能性が高いため、可能であればテストとセットで作成する方が望ましいが、適切なフィードバックのためには「学習者がどうしてそう答えたか」という誤答分析が先行する必要がある。図 1 に示すようなそもそもテスト機能を利用していない環境では誤答分析に用いる情報自体が蓄積されていないため、高機能な作問システムがあっても有効利用は難しい。さらに、特定の LMS へのサポートを強めると、他の

16　https://docs.moodle.org/34/en/Aiken_format　より引用

LMS へのポータビリティが減少する問題も大きくなる。

2.3.2. 特定機能に限定して作問サポートは使いづらい

本論文では Moodle の作問支援システムをサーベイ対象としたため、穴埋め問題の作問支援システムが多く取り上げられた。しかしながらこれは Moodle の穴埋め問題を GIFT 形式で作成すると、問題作成者にとっては非本質的と考えられる大量の情報を記載する必要があるという Moodle 固有の問題であり、LMS 一般に穴埋め問題限定の作問支援システムが必要であるとは考えにくい。また、作問時に問題形式によってツールを変更する必要があるのは不便であり、問題管理の都合上からもあまりに限定的な問題作成システムは使いづらいと考えられる。

2.3.3. 機能をできる限り限定して、楽に作問できるようにする

Aiken フォーマットと Missing Word フォーマットはそれぞれ単一選択問題、穴埋め問題のみに対応するフォーマットであるが、普通のエディタで簡単に記述できることと、制御情報をほとんど含まないために紙に出力して試験に流用することも比較的容易である。ただし、Aiken フォーマットと Missing Word フォーマットは単一の問題形式にしか対応しないため、特定機能の支援ツールと同様に問題管理上の煩雑さが出てくることと、図版などの組み込みがサポートされていないため作成できる問題の表現力に問題がある。

2.3.4. 望ましいテスト作成方法

本論文では、上記の考察より以下のようなテスト作成方法を提案し、dotCampus 用に実装する。

1．穴埋め、単一選択。複数選択問題を 1 つの問題ファイルに混在できる。
2．ユーザの記述量をできるだけ減らす。パラメータについては記述がなければデフォルト値を採用する。
3．図や数式は使えるようにする。

3．小テスト記述言語の定義と処理システムの実装

よく使われる問題タイプとして穴埋め、単一選択、複数選択の 3 種類を一括して記述できるようなファイル形式を定義する。発想としては、Missing Word 形式と Aiken 形式で作成できる問題タイプを、単一のファイルに統一的な形式で記述できるようにすることを目指している。

3.1. 言語定義

3.1.1. 問題ファイル[17]

問題ファイルは複数の問題を含むテキストファイルである。テキストファイルにしておくことで、作成ツールには慣れたエディタ、問題の管理には Git などの汎用リビジョンコントロールシステムを用いることができる。テキストファイルの文字コードはシフト JIS あるいは UTF-8 を用いる[18]。問題ファイルのファイル名は source.txt で固定。

記述は行単位で行い、行頭に # とアルファベット 1 文字からなるコマンドを埋め込むことで指

17　論文執筆後の更新も含めて、ファイルフォーマットに関する最新情報は https://eip.econ.kanagawa-u.ac.jp/campus 5 /source.html　にまとめられている。

示を行う。問題と問題の区切りは、1行以上の空行を入れることで行う。

3.1.2. 問題ファイル全体に関わる設定

#T タイトル[19]

複数の問題を一括して扱う名前を定義する。dotCampus の場合、ユーザから見える問題セットの名前はここで設定した名前だけとなるため、適切な名前を付けることが重要。

3.1.3. 個々の問題に関する指示

#Q 問題文（Question）

単一選択、複数選択問題の場合：問題文を指定する。#Q は1つの問題に何行出てきても構わないが、#Q で区切った行ごとに出力時は改行する。積極的に改行する必要がなければ、1つの #Q に長く書いて表示側の画面幅に応じた処理に任せた方が無難。

穴埋め問題の場合：穴埋めの対象となる問題文を指定する。穴埋め問題を作成する際は、穴を開けたい部分を [　]（角カッコ）で囲む。正解文字列については、[　] 内の文字列を自動的に利用する。

#C 見出し（Caption）

穴埋め問題で問題文以外の指示を書くために利用する。穴埋め問題でしか使えないため、選択肢を表す行（後述の #A や #S）が入っている問題に同時に #C が指定された場合はエラーになる。

#A 正解選択肢

単一選択問題、あるいは複数選択問題での正解選択肢を書く。1つの問題につき、最低1つは必要。#A が1つならば単一選択問題、2つ以上あれば複数選択問題として自動的に数を数えて処理する。

#S 誤答選択肢

単一選択問題、あるいは複数選択問題での誤答選択肢を書く。1つの問題につき0個以上必要。#S が存在しない（＝全て正しい選択肢）という作問も可能。

#G 解説文

個別の選択肢に対応するフィードバックを記述することはコストが高いため、本システムでは問題全体に対する解説を書くことができるようにした。オプショナルな部分で、#G が存在しない問題があっても特に問題はない。

#P 配点

問題の配点。デフォルトでは1点が割り当てられている。必要がなければ特に指定する必要はない。

#I イメージファイル指定

問題文と一緒に表示される画像ファイルを指定する。画像ファイルは、source.txt と同じ zip ファイルに圧縮してシステムにアップロードする。

#F 添付ファイル指定

18　Windows のメモ帳で作問した場合のデフォルト文字コードがシフト JIS であるため両方受け付けるように仕様上は決めているが、今回作成した変換システムでの最終出力先は dotCampus 仕様の XML であり、出力コードは UTF-8 となる。そのため、シフト JIS を用いている場合はコード変換処理による文字化け（特に記号類）が発生する可能性がある。

19　タイトル行には dotCampus 固有の属性情報（科目名や年度など）も入力できるが、省略。

図 4　教材ライブラリへの問題セットインポート

問題のヒントや解き方の情報を追加のファイルで与えたい際に使うファイル指定。たとえば、計算問題で、計算式が入っている Excel ファイルをダウンロードさせるなどの利用方法を想定している。

3.1.4.　dotCampus を前提とした拡張機能

dotCampus サーバーではない別サーバーに置いた画像を強制的に表示させる方式で実装している。**#C, #Q, #S, #A, #G** の中でだけ有効。

インラインでの画像表示

¥f{ 画像ファイル名 } を挿入した部分に、画像ファイルを表示する。画像ファイルは、source.txt と同じ zip アーカイブに入れる。

数式の挿入

¥t{ pLaTeX コマンド } を挿入した部分に、記載した pLaTeX コマンドから生成した JPEG イメージを表示する。使える pLaTeX コマンドはセキュリティ上の理由から大幅に制限されているため、難しいことをする場合はローカルで画像まで生成して ¥f{ } で張り込む方が確実。

文字修飾

¥i{}, ¥b{}, ¥u{} で囲んだ文字列に、それぞれイタリック、ボールド、アンダーラインの文字修飾を行う。

外部リンク

¥l{url| リンク対象文字列 } を挿入した部分に、外部 URL に対するリンクを埋め込む。

3.2.　dotCampus を対象としたシステム実装例

dotCampus は問題セットを XML ファイル＋関連するメディアファイルをアーカイブした zip ファイルとしてインポートすることができる（図 4）。そこで、上で定義した小テスト記述言語を用いて作成した問題を dotCampus の問題インポート用の XML ファイルに変換するシステムを実装してみた[20]。

20　https://eip.econ.kanagawa-u.ac.jp/campus 5 /

図 5　問題変換の流れ

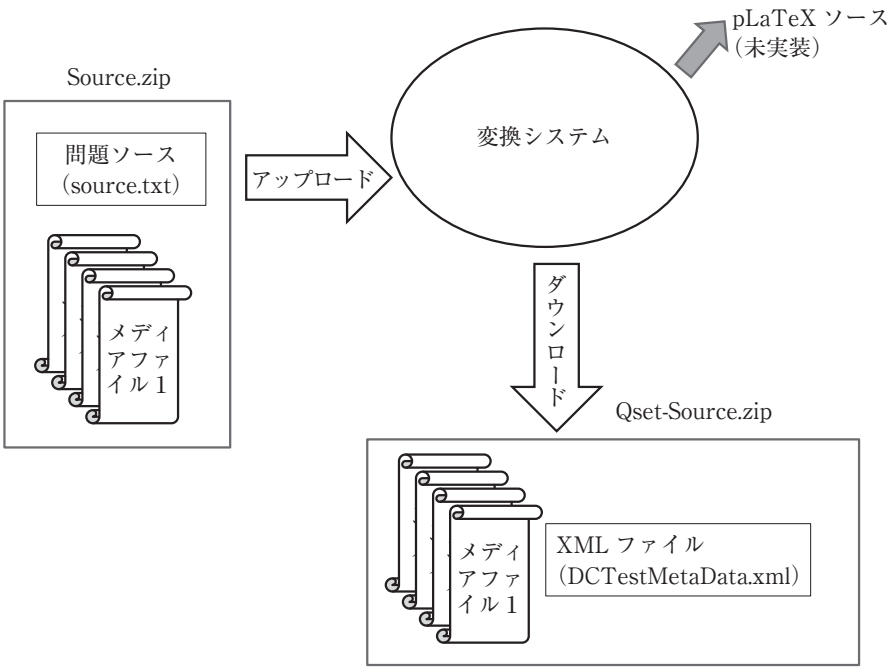


図 6　問題ファイル（source.txt）例

```
#T 小テスト問題

#Q 図はある市場での需要、供給の状態を示している。この市場において価格 \i{p} では、どのようなことがおこっているか選びなさい。
#A 超過需要 > 0である
#S 超過需要 < 0である
#S 超過価格が存在している
#S 超過市場が存在している
#G ある価格に対応する需要曲線上の点（需要量）と、供給曲線上の点（供給量）を比べて供給量が多いときは超過需要はーとなる。
#I figure 1 .jpg

#C GDP 成長率に関する以下の文章の空欄を適切な数字でうめなさい。ただし数字は半角で入力すること。
#Q ある国の前年の GDP が150兆円であった。この国が前年に比べて 4 % の経済成長を遂げた場合、この国の今年の GDP は [156] 兆円である。
#G 前年比の経済成長率は、¥t{¥frac{ 今年の GDP − 前年の GDP}{ 前年の GDP}} で定義される。これを今年の GDP について解くと、¥t{ 今年の GDP = ( 1 + 前年比の経済成長率 ) ¥times 前年の GDP} となる。この式に値を代入すると156兆円が得られる。
```

システムの大まかな構造は、図 5 に示す通りである。ユーザはローカルでテキストエディタ（メモ帳など）を用いて問題ファイルを作成し、必要なメディアファイルと同一の zip アーカイブに入れる。そして、そのアーカイブを Web 上で実装されている変換システムにおくると、dotCampus にアップロード可能な zip アーカイブに変換される。図 4 での「XML ファイルから追加」で指定するファイルが、図 5 での Qset-Source.zip になる。

図 7　問題に挿入する図版の例（figure 1.jpg）

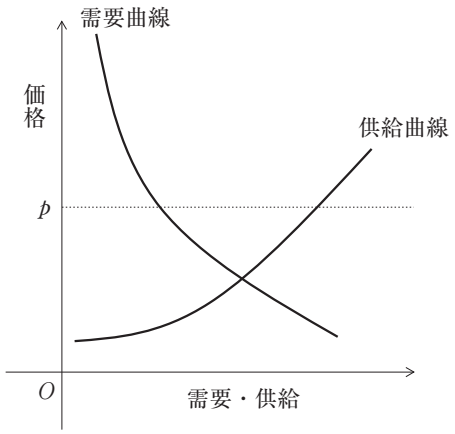


図 8　dotCampus XML に変換したデータ（一部）

```
<?xml version="1.0" standalone="yes"?>
<NewDataSet>
<Test>
<TestID>00000000-0000-0000-0000-000000000000</TestID>
<TestName> 小テスト問題 </TestName>
<ContentNote />
<Year>2017</Year>
<CourseName />
<Tags>
</Tags>
<Question>
<OrderNum> 1 </OrderNum>
<QuestionText> 図はある市場での需要、供給の状態を示している。この市場において価格 &lt;i&gt;p&lt;/
i&gt; では、どのようなことがおこっているか選びなさい。</QuestionText>
<Type> 1 </Type>
<QuestionText 2  />
<Option>
<OptionText> 超過需要 &amp;nbsp;&amp;lt;&amp;nbsp; 0 である </OptionText>
```

3.3.　実際の変換例

穴埋め問題と複数選択問題を含む単純な問題ファイル（図 6、図 7）を例に取って、実装してみたシステムでの変換を行った結果を示す。変換結果の XML ファイルは（図 8、部分）であり、この問題を dotCampus に読み込んでテストとして作問した場合は（図 9）のように表示される。

3.4.　dotCampus 内蔵問題エディタとのパフォーマンス比較

dotCampus 内蔵エディタで図 6 と同じ問題を作成するためにはテキストをコピー＆ペーストする他にマウスクリックとそれに伴う画面遷移が30回以上必要であった。このクリック数は問題を作成する数が増えるとリニアに増加するため、１問につき少なくともクリック15回以上のパフォーマンスの差が存在することになる。

図 9　dotCampus での表示例

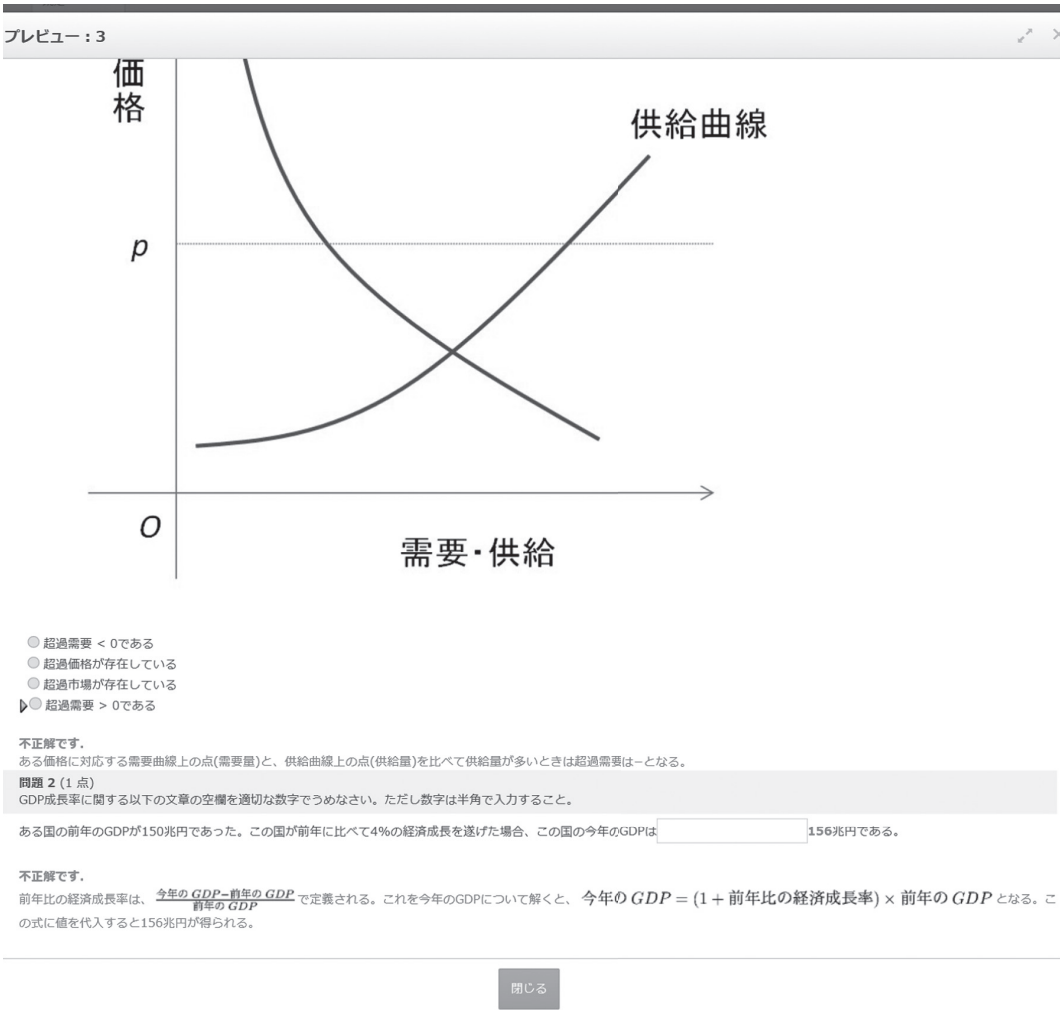


図10　問題ソースから生成した紙テスト（マークシート回答用）の例

- わが国の派遣に関する記述で正しいものをすべて **問 9** にマークしなさい。

  ① 現在の法律では、専門 26 業務は期間制限なし、それ以外の一般派遣は最長 3 年まで派遣可能である

  ② 派遣が法的に認められたのは小泉内閣の 2004 年になってからである

  ③ 派遣は当初専門的な 13 業務に限定して許可された

  ④ 現在の法律では、派遣はいくつかの禁止業務以外は許可されている。このような方式をネガティブリスト方式と呼ぶ

3.5. 紙テストへの応用

今回実装したシステムの範囲にはまだ含まれていないが、今回の作問言語仕様に沿って作成した問題ファイルから pLaTeX ソースを作成して、マークシート用の問題を作った例を（図10）に示す。現時点ではこのシステムはスタンドアロンアプリケーションとして実装されているが、今後、今回作成したシステムのサブシステムとして、PDF および pLaTeX ソースのダウンロード

可能なシステムを実装予定である。

4．まとめ

本論文で提案した小テスト記述言語は以下の利点を持っており、先行の各種フォーマット、ツールよりも「簡単に沢山問題を作る」という用途を考えた場合には有効である。

1．記述が簡単である

普通のテキストエディタで記述可能であり、なおかつできる限りユーザが制御用に記入する部分は少なくなるように設計されている。

2．表現力はある程度ある

同様に単純なテキストファイルで記述できる Missing Word フォーマットや Aiken フォーマットと比較すると、穴埋め、複数選択、単一選択の問題が同一ファイルで作成可能であり、さらに数式や図も挿入可能と表現力にも配慮されている。

3．余計なことをしない

dotCampus の内蔵エディタで同様の問題を作成する場合、インタラクティブな操作を求められるため問題数が増えるとリニアにクリック数が増えて、作業時間も増えていく。本論文で提案したフォーマットは、紙での出題形式に僅かな制御文字列を追加するだけであり、記載にかかる手間は圧倒的に少ない。また、簡単なテキストファイルであるため Excel や Python などを使って自動生成することも可能である。計算問題などは数字を変えて自動生成することで、練習問題を容易に作成可能となる。

4．流用可能である

今回実装した Web 上のシステムには含まれていないが、同一のソースから紙テスト用の問題を作成することも可能である。たとえば授業後のポストテストは LMS で行うが、成績評価用の小テストは紙で行う、といった使い分けをする場合には、このような流用可能性が担保されていることは重要であると考える。

●引用文献

ディック，ウォルター＆ケアリー，ルー＆ケアリー，ジェイムズ，O.『はじめてのインストラクショナルデザイン』．翻訳者：角行之．ピアソン・エデュケーション，2004.

鈴木克明.『教材設計マニュアル』．北大路書房，2002.