DP レポート

オンラインプログラミング学習・試験システム track

新田章太（（株）ギブリー）　小西俊二（（株）ギブリー）
竹内郁雄（東京大学名誉教授，（株）ギブリー技術顧問）

概要　企業におけるプログラミング研修や，学校におけるプログラミング教育では，プログラミング言語の多様化や高度化，頻繁なアップデートへの対応が常に必要である．このためには，少ないヒューマンリソースで，学習教材・試験の編集・改善を行えることが望まれる．オンラインプログラミング学習・試験システム track は，複数のプログラミング言語に対応しやすい実行環境や自動採点機能を搭載したシステムで，オンラインプログラミング教材や問題を誰もがスムーズに作成・編集できる．

1. はじめに

多様な教育現場へのプログラミング教育の普及に伴い，プログラミング学習環境の改善が従来多く取り上げられてきた[1], [2], [3], [4], [5], [6], [7], [8]．例えば，Bit Arrow に代表される，学校における従来のプログラミング授業を効率化するための，Web ブラウザを用いたオンライン学習環境の提供がある．

しかし，時代に適合したソフトウェア開発を必要とする企業におけるプログラミング研修や，新しいプログラミング教育法を開拓している学校では，Java，Ruby，Python など，プログラミング言語の多様化や高度化，言語改訂へのタイミングのよい対応が必要になる．

我々は，学校のみならず企業でも利用できる track というオンラインプログラミング学習・試験システムを開発した．track はコア部分とコンテンツ層，アプリケーション層を分離し，多様な利用環境とプログラミング言語に対応したオンライン教材・試験を誰もがスムーズに作成・編集できるシステムである．本稿では track が解決すべき課題とそれを解決するための実装手法について述べ，適用事例についても簡単に紹介する．

2. 課題

2.1 実行環境・教材の更新

プログラミング言語の進化は，型推論の導入などの便利な機能の充実だけではなく，それまで標準とされていた記法を変えることがある．例えば，Java は 1995 年以来，バージョンアップを重ねており，ORACLE 社は今後 6 カ月に一度のメジャーアップデートをしていくと発表した[9]．教材管理は，半年スパンで発生するこのような変化に対応し続けなければならなくなっている．

2.2 実行環境・教材の更新

プログラミング言語の多様化も進んでいる．プログラミング言語はそれぞれのコミュニティや管理者を通じて更新され，応用分野の変化に対応した新しい技術トレンドが生まれている．

社会の現場で利用されている，あるいは注目度や必要性が増加するであろうプログラミング言語を教育に採用したい場合，時代のトレンドを掌握し，時宜にかなった教材を作成するのに担当者の大きな手間が必要となる．

2.3 採点にかかるヒューマンリソース

これまでのプログラミング授業における演習や試験では，受講者や受験者（以降，まとめて受講者と呼ぶ）が PC で書いたプログラムを，メールや，授業管理システム，USB メモリなどを通じて提出し，問題作成者あるいは TA がそれらを個別に確認・採点していた．これにも大きな手間が必要である．

3. track 開発の経緯

我々は 2013 年 8 月からオンラインでプログラムを書いて，動かしながら学べる実践型のプログラミング学習サービス CODEPREP（コードプレップ）を開発し，2014 年 8 月に無料公開した．現在 8 万人が登録する国内有数のオンラインプログラミング学習サービスである．現在でも教材をクリアした多くの受講者が気軽に Twitter などの SNS で発信している．

しかし，第 2 章で述べたように，公開している教材を言語のバージョンに合わせて更新する必要や，多言語の実行環境をサポートするために莫大なコストと時間を要することが問題となり，それらを簡単に管理する仕組みが求められた．

そこで，複数言語の実行環境に対応し，教材を簡単に作成・変更ができるエンジン（後述の track コア）を開発した．この仕組みを用いて，企業が採用時や研修時に受講者のプログラミングスキルを把握するためのスキルチェックツールとして，track の前身となる codecheck（コードチェック）を 2015 年 10 月に公開し，50 社近くの企業へ展開してきた．

前述のプログラミングの学習における課題を解決し，これまで両軸で展開していた 2 つのサービスを，1 つのプラットフォームとして統合し，発展性を持たせたのが track である．2018 年 3 月に初版を公開した．

4. track のコンテンツと機能

4.1 track が扱えるコンテンツの種類

track が扱えるコンテンツ，すなわち問題や教材は以下のものがある．

チャレンジは，目安となる制限時間の中での解答を採点する問題で，授業後の演習などに使う．チャレンジを複数個選択して束ね，明確な制限時間あるいは提出締切をつけたものが**試験**で，期末試験や採用試験などに活用できる．

チャレンジには，コーディング形式とクイズ形式がある．

与えられた要件を満たすプログラムを書くのがコーディングである．受講者が解答に利用したいプログラミング言語を選択すると，言語ごとのテンプレートが

与えられるので，それに必要なコードを書き加える．それぞれの問題には，要件が満たされていることを確かめる，ユニットテスト形式で書かれたテストケースが 10〜30 個（一部は受講者に非公開）用意される．このため，満点以外の部分点が取れることになる．

ラジオボタンやチェックボックスで 4 択の問題に解答する，あるいは空欄を文字や文字列で穴埋めするのがクイズである．

ブックでは，受講者が複数の章と節から構成される平易なチャレンジ集を，解説文や問題文を読みながら解答していく．クイズを解き，プログラムを書いて実行しながら体系的にプログラミング言語に習熟できる．ブックを複数個，系統的に配置し，最後に少し重めのチャレンジを加えたものを**コース**と呼ぶ．

ブックには，ドリル形式と演習形式がある．

問題文とプログラムの一部が空欄で与えられ，適切な解答を選択するか，穴埋めすることで，プログラムが実行され，次に進むことができるのがドリルである．文章や図のみで構成されている解説テキストの節を追加して，読むだけの教材にすることも可能である．

問題文とプログラムが与えられ，プログラムを編集・実行し，すべてのテストケースを通すことで，次に進むことができるのが演習である．

4.2 track の 3 つの主要機能

ブックやチャレンジを作成し，コースや試験を組み立てるだけでなく，受講者がそれらを受講・受験できるようにすること，また採点やレポート作成をすることも重要な機能である．本節の**表 1** に track と国内の類似システムとの比較をまとめた[8],[10],[11].

表 1　国内の類似システムとの比較

	Bit Arrow	GAIT	TOPSIC	track
コンテンツの種類	コーディング教材	選択式問題	アルゴリズム問題	多種類の教材･問題
コンテンツ作成環境	○	×	×	○
配信管理	学習	試験	試験	学習･試験
多言語実行環境	5 言語	×	主要言語	主要言語
自動採点	×	○	○	○

4.2.1 コンテンツ作成機能

track のブックやチャレンジは，GitHub 上で管理される，コンテンツの形式ごとにまとめられたテンプレートをベースに，テキストや問題文，模範解答コード，テストケース，詳細設定などを編集して作成し，ブックやチャレンジとして登録する．登録されたブックとチャレンジは一覧可能で，それらを組み合わせて，目的に合わせたコースや試験を作成する．

4.2.2 コース受講・受験機能

作成したコースや試験は受講者にメール配信する，あるいは専用 URL に受講者をアクセスさせることで届ける．受講者はそれぞれの方法によって受け取った URL に Web ブラウザからアクセスして，コースや試験を受ける．

図 1　track エディタの画面

試験の場合，受験者は Web ブラウザエディタ（**図 1** の track エディタ）を利用して提出締切までに問題を解く．コーディング問題では，プログラムを実行し，エラーメッセージを見ながら修正することができる．編集・保存が終わったら，提出ボタンを押して，解答を提出する．後述するように，受講者がプログラミング言語を自由に選択して解答できることが track の特徴である．

4.2.3 採点・レポート機能

コースの進捗状況や，提出された解答と採点結果は，全体のレポートと，受講者ごとの個別レポートとしてまとめられる．解答データや得点を見やすい形で可視化することで，企業研修の効果測定や授業内容の理解度チェック，受講者の評価に利用することができる．

5. track の構造と実装

本章では，第 2 章で述べた課題を実装面でどのように解決し，第 4 章で述べた機能を実現したかについて述べる．track は対応言語を増やしやすくするためと，利用環境・状況に応じて授業や試験の管理を柔軟に設計できるようにするために，track コアと呼ぶエンジン部分とそのほかの部分を 2 層にして分離したことが特徴である．track は次の 3 個の層からなる．

- track コア：track の共通基盤であり，以前エンジンと呼んでいたもの．
- コンテンツ：4.1 節で述べたコンテンツ群（ブックとチャレンジ）の層．
- ソリューション：track をいろいろな環境や状況で利用するための管理アプリケーション群．例えば，コンテンツ群から配点や時間配分を決めた試験を作成し，その配信・採点・レポート作成などを行ったり，大学の授業で使うブックをまとめたりするのはこの層である．

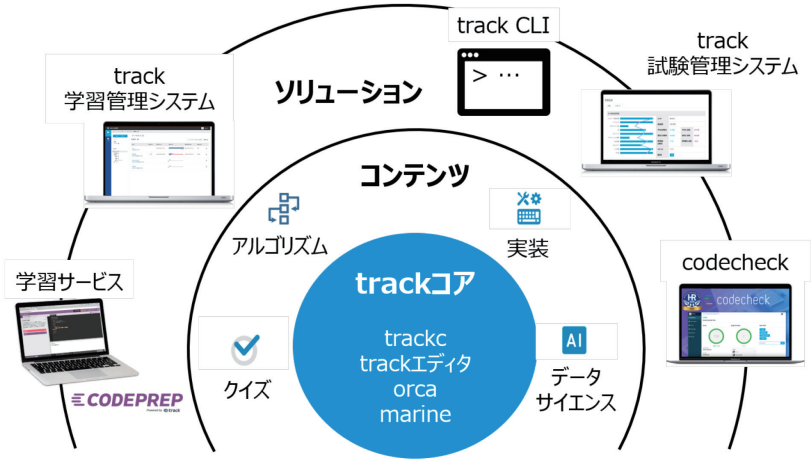

図 2　track の 3 つの層

図 2 の track コアから広がる同心円が track の全体構造である．過去に開発した CODEPREP や codecheck もこの同心円の外周部に乗っている．

5.1 track コア

track コアは次の 4 個のコンポーネントからなる（**図 3**）

- track コンパイラ trackc
- track エディタ
- コード実行サーバ orca
- コンテンツ作成ツール marine

5.1.1 track コンパイラ trackc

コンテンツ作成は，簡潔で安定した環境を維持するためにテキストエディタのみで可能とした．trackc はコンテンツのソースである Markdown 言語[12]で書かれたテキストを track エディタで実行可能なパッケージに変換するモジュールである．Markdown には画像などを含めることもできる．

5.1.2 track エディタ

track エディタは trackc が変換したコンテンツをブラウザ上で表示・実行するフロントエンドである．このモジュールには，コンテンツがどのように供給されるか，受講者の解答がどのように管理されるかを規定する機能は含まれておらず，コンテンツを表示・実行する機能だけを受け持つ．track エディタ上で編集したソースコードは後述のコード実行サーバ orca に送られ，その実行結果（標準出力）が画面上に表示される．通常，track エディタは外側のシステムから html の iframe として実行され，コンテンツ供給を行うのはソリューション層の役割である．

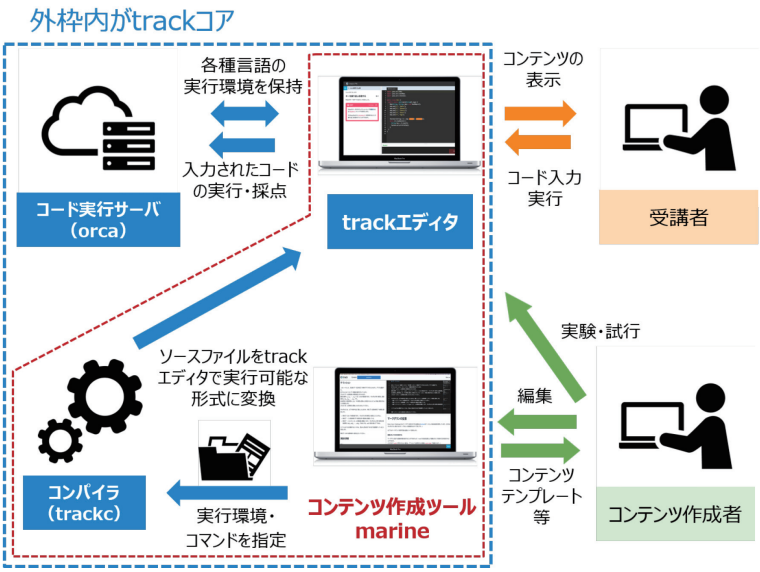

図 3 track コア

5.1.3 コード実行サーバ orca

orca は受講者が記述あるいは穴埋めしたプログラムを実行するバックエンドサービスである．orca は，受講者が記述したソースコード，コンテンツ作成者が提供するテストケースで使用するファイル，および実行するコマンドを受け取り，用意されたコンテナシステム Docker[13]をその都度起動して，その中でコマンドを実行する．コマンドはコンテナ内で実行されるので，受講者が悪意のあるコードや不用意な無限ループを実行したとしても影響範囲はコンテナ内のみである．

コマンドを実行した結果，プログラムに構文エラーがあれば，当該言語処理系のメッセージをそのまま標準出力に出す．無事実行可能となれば，その結果が正

しいか，間違っているかの検査を行う．これは orca 内の検査モジュールが行う．穴埋め問題の解答の正誤チェックは正解に幅を持たせるため正規表現で行う．コーディング問題の解答の正誤検査はテストケースを用いて行う．

5.1.4 テストケース

コーディング問題の作成者が行う最も重要な作業はテストケースの作成である．入力データやパラメータに対応する出力値を用意することでテストケースとする．

テストケースは，Java → Junit，Ruby → RSpec，Node.js → mocha のように，それぞれの言語における代表的なフレームワークを使用して記述することができる．それぞれのテストフレームワークの出力を TAP 形式[14]に揃えることによってテストケースの成否を透過的に取得することができる．これは多言語対応のための重要な手段の 1 つである．

テストケースをベースとした採点方式は TDD (Test-Driven Development) の考え方[15]を応用したもので，実際の業務との親和性が高い．

テストケースには受講者に見えないものもある．つまり，受講者視点からは 10 個のテストケースがあって，そのすべてにパスしているように見えても，隠されたエッジケースが存在して，そこで採点に差がつくことがあり得る．

コーディング問題のテストケースは track エディタで検査されるほか，後述のソリューション層の track CLI や試験採点モジュールからも検査される．

5.1.5 CLI による言語非依存のテストケース

前節で述べたような言語毎のテストフレームワークを用いてテストケースを検査すると，受験可能な言語が限定されてしまう．しかし，アルゴリズムを問う問題では受講者が任意の言語を選択できることが望ましい．これを実現するために CLI (Command Line Interface) を扱う独自のテストフレームワークを用意した．

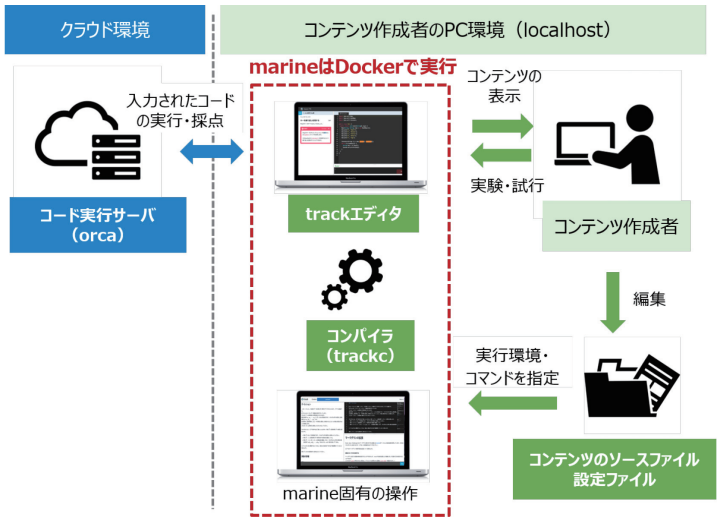

図 4　コンテンツ作成の仕組み

このフレームワークにより，受講者は track がサポートしている言語なら何を使用してもよく，標準入力からパラメータやデータを受け取り，標準出力に結果を出力する CLI プログラムを作成すればよい．また，言語ごとに標準入出力を扱うテンプレートが用意されているので，受講者はアルゴリズムの実装に集中できる．この方式で作成されたチャレンジをアルゴリズムチャレンジと呼ぶ．

5.1.6 コンテンツ作成ツール marine

marine はコンテンツ作成者に提供されるチャレンジとブックの作成用アプリケーションである．コンテンツは，テキストベースであるため GitHub 等の既存の VCS (Version Control System) でソースを管理できる．

marine は Docker イメージとして提供されており，コンテンツ作成者は自分のローカル環境で実行することができる．Docker コンテナは trackc と track エディタを含む（**図 4**）．コンテンツ用のディレクトリは，ホスト側のディレクトリとリンクしておりコンテンツ作成者はローカル環境上でのコンテンツの編集が可能である．marine は，コンテンツディレクトリから一連のソースファイルを読み込み，それを trackc でコンパイルして，コンパイル結果を track エディタで表示する．このためコンテンツ作成者は実際の画面での表示を確認しながら，コンテンツを修正できる．track エディタ上で編集した内容を保存しないで orca で実行することもできる．これはコンテンツ作成段階での実験を容易にする．

5.1.7 ブックの作成

ブックは Markdown を拡張した専用のブック記述言語 BMD で記述する．アプリケーションの作成法を解説するようなブックでは，前節の内容を少しだけ変更・追記するという内容のページが多く発生するが，こうしたブックの作成を容易にするための組み込み関数を BMD に多数用意した．

5.2 ソリューション層

ソリューション層のシステムは，track コアの上に，CLI を使ったローカル受験の機能，コンテンツ作成者の作成したチャレンジやブックを管理する機能，受講者の解答コードを管理する機能，受講者の解答コードを orca で実行し，その採点を管理し，可視化する機能などを加えてまとめたものである．ソリューション層での変更は track コアには影響を与えない．これは track コアを利用して多様なソリューションが作成できることを意味する．Track 以前の codecheck もその 1 例であるが．以下では track CLI について簡単に紹介する．

コンテンツの解答は通常 track コアの track エディタ上で行うが，コーディング問題に関しては track CLI というコマンドラインアプリケーションを使用することで，受講者は自分のローカル環境で使い慣れたエディタを利用して解答することができる．track CLI は Windows, Mac, Linux の環境で動作する．

6. システムの拡張性

Docker コンテナの徹底的利用は track の拡張性に大きく寄与している[17]．例えば，orca は Docker をベースに構築されている．このため，各言語の Docker イメージさえ準備すれば様々な拡張が可能となる．

6.1 対象言語の新規追加

例えば，track は本稿執筆時点ではまだ Haskell 言語をサポートしていないが，Haskell を追加するための作業は，Haskell 用の Docker イメージを用意してコード実行サーバ orca に含め，アルゴリズムチャレンジ用のテンプレートを用意するだけである．そのためサポート言語が容易に追加できる．

実際，2019 年 7 月に Kotlin，Swift 言語のサポートを追加したが，それぞれ正味 1 人・1 週間で追加できた．

6.2 対象言語・フレームワークのバージョンアップ

Java は半年に一度，Ruby はほぼ年に一度，言語のバージョンアップが行われている．他の言語もそれに近い周期でバージョンアップされるのが一般的である．また，mocha や RSpec のようなテストフレームワーク自体がバージョンアップされることもある．

これに対応する方法は，対応する Docker イメージの差し替えである．track において受講者が書くコード量はそれほど大きくないので，互換性を重視する言語であれば，旧バージョンを新しいバージョンに置き換えてもほとんど問題にならない．ただし，厳密に過去の実行環境を残す必要がある場合は，置き換えではなく新しい Docker イメージの追加で対応することになる．

6.3 track の拡張の実際

track は，実行環境をサポートしているバックエンドのプログラミング言語が 2020 年 10 月末時点で，C, C++, C#, Node.js, Java, Perl, Ruby, Go, Python2, 3 系，PHP, Scala, Kotlin, Swift, Rust と，人気実用言語の多くをカバーしている．拡張しやすいので，リクエストがあれば簡単にサポート言語を増やしていける．

また，track では，HTML と Javascript に加え，CSS 言語のブックを用意している．このようなフロントエンド言語の場合は受講者の記述が即時に track エディタ上のプレビューに反映される．

7. track の利用実践

track は 150 社以上の採用試験や研修に導入されており，そのうちの約 20%の企業の採用・研修担当技術者が，track コアを利用して独自のコンテンツ作成を行っている．

また，2019 年度度後半から，大学・高専教育への track の無償提供を行っており，すでに 9 校に導入された．中でも，東京大学工学部の 3 年生（4 年生と大学院生を含む）160 名を対象とした 2020 年春学期のリモート講義の演習に利用した矢谷浩司準教授には独自のアルゴリズムのブックと試験を作成していただいた．学生アンケートでも高い評価を得た．

青山学院大学の松澤芳昭準教授は，文系初学者 300 名向けに HTML, CSS, JavaScript を用いた Web アプリケーションのブックを作成，津田塾大学の来住伸子教授，大塚亜未助教は理系初年級の入門教育第 1 四半期・第 3 四半期（50 名）に Java を用いたチャレンジを作成し，それぞれ授業にて活用していただいた．

これらの実績からのフィードバックは track の改良に順次反映されている．

8. おわりに

本稿では現在のプログラミング教育環境における課題の定義と，言語・技術アップデートや多言語に対応しやすい track のシステム構成について述べた．コアとそれ以外の分離，Docker の利用などが大きな特徴である．track を利用している企業の採用試験では，国内だけでも，我々の提供したコーディング問題に対して年

間 15000 人以上の受験者の約 3 万個の解答が集積している．それを分析することにより興味深い知見が得られる見込みである．

謝辞 会津大学渡部有隆上級准教授から，Aizu Online Judge（AOJ）に掲載しているオンライン学習教材を提供していただいた．

参考文献

1) 井垣 宏，齊藤 俊，井上亮文，中村亮太，楠本真二: プログラミング演習における進捗状況把握のためのコーディング過程可視化システム C3PV の提案，情報処理学会論文誌 54(1), 330-339, 2013-01-15.
2) 長島和平，本多佑希，長 慎也，間辺広樹，兼宗 進，並木美太郎: オンラインで複数言語を扱うことができるプログラミング授業支援環境，情報教育シンポジウム 2016 論文集 2016, 137-140, 2016-08-15.
3) 長島和平，長 慎也，間辺広樹，兼宗 進，並木美太郎: Web ブラウザを用いたプログラミング学習支援環境 Bit Arrow の設計と評価，研究報告コンピュータと教育（CE），2017-CE-138, 2, 1-8, 2017-02-04.
4) 間辺広樹，長島和平，並木 美太郎，長 慎也，兼宗 進: オンラインプログラミング学習環境 Bit Arrow を用いた JavaScript の授業実践報告，研究報告コンピュータと教育（CE），2017-CE-138, 3, 1-8, 2017-02-04.
5) 齋藤宏太郎，豊田哲也，大原剛三: オンラインプログラミング学習システムのための適応型出題モデルの提案，第 79 回全国大会講演論文集，2017, 1, 685-686, 2017-03-16.
6) 間辺広樹，長島和平，並木美太郎，長 慎也，兼宗 進: 自宅で行うオリジナル作品制作の学習効果と問題点 ～オンラインプログラミング学習環境を用いて～ 情報教育シンポジウム論文集，2017, 15, 101-109, 2017-08-10.
7) 岩田麻暉，長島和平，長 慎也，兼宗 進，並木美太郎: プログラミング学習環境 Bit Arrow における JavaScript によるデータベース API の設計と実装，研究報告コンピュータと教育（CE），2018-CE-143, 3, 1-9, 2018-02-10.
8) 長慎也，長島和平，間辺広樹，兼宗進，並木美太郎: オンラインプログラミング環境 Bit Arrow における Python 処理系，情報教育シンポジウム 2019 論文集 2019, 122-129.
9) Donald Smith: Faster and Easier Use and Redistribution of Java SE, https://blogs.oracle.com/java-platform-group/faster-and-easier-use-and-redistribution-of-java-se
10) https://www.gait.org/
11) https://products.sint.co.jp/topsic
12) https://daringfireball.net/projects/markdown
13) https://github.com/docker/docker-ce
14) https://testanything.org/
15) Beck, Kent (2002). Test-Driven Development: By Example. Addison-Wesley Professional. ISBN 0321146530, [邦訳] ケント・ベック『テスト駆動開発入門』長瀬嘉秀監訳，（株）テクノロジックアート訳、ピアソン・エデュケーション，2003 年，ISBN 4894717115.

新田章太（正会員）shota.nitta@givery.co.jp
1989 年生．2012 年 筑波大学理工学群社会工学類卒業．同年株式会社ギブリー入社．2014 年同社取締役就任．オンラインプログラミング学習・試験システム track のプロダクトオーナー．

小西俊二（非会員）shunji.konishi@givery.co.jp
{1973 年生．1996 年横浜国立大学経営学部国際経営学科卒業．SI 企業，ミドルウェア開発企業を経て 2015 年より株式会社ギブリーにてオンラインプログラミング学習・試験システムに従事．

竹内郁雄（正会員，フェロー）nue@nue.org
1946 年生．1971 年東京大学理学系研究科数学専攻修士課程修了．博士（工学）．1971 年 NTT 研究所．1997 年電気通信大学，2005 年東京大学，2011～2013 年早稲田大学の教授を歴任．2000 年より IPA 未踏事業（統括）プロジェクトマネージャ．2016 年より株式会社ギブリー技術顧問．

投稿受付：2020 年 11 月 30日
採録決定：2020 年 12 月 9 日
編集担当：新田 清（Yahoo! JAPAN 研究所）