第14回情報システム教育コンテスト（ISECON2021）

遠隔演習環境の開発と運用を通したプログラミング教育の改革

福岡大學

工学部電子情報工学科

中西 恒夫（福岡大学），秋山 英久（岡山理科大学），請園 智玲，田辺 利文，中村 遼，橋本 浩二，藤永 拓矢（福岡大学），古庄 裕貴（元・福岡大学），前田 佐嘉志，三角 真，杨 同鑫（福岡大学），久住 憲嗣（芝浦工業大学）

本取組みの背景

2020年春に始まるコロナ禍により対面によるプログラミング演習ができなくなった。

- 学生はオンライン／オンデマンドで自宅で自助努力を伴う演習を強いられるようになった。
- ほぼすべての学生がPCを保有しネット接続可能な状況だが，学生個人のPC＆ネット環境は不揃い，場合によっては貧弱だった。
- 「スマホ世代」の学生のPCリテラシは概して高くなく，プログラミング演習以前にPC環境の操作等のトラブルシューティングでしばしば演習が滞る。（OSやシェルの操作，ファイルシステムの知識，ソフトウェアのインストールなど）
- 演習科目の履修生は70～100名と大人数だった。
- コロナ禍前より演習に割り当てられる教員リソースは逼迫気味だった。

【危機感】オンライン／オンデマンド講義でプログラミング演習を円滑，かつ十分な品質で実施することが危ぶまれた。

本取組みの概要とねらい

概要：遠隔プログラミング演習のためのサーバ／クライアントシステムを開発・運用し，遠隔かつ数十人規模でも，円滑にプログラミング演習を教員チームで実施した。

- 軽量開発環境（リッチクライアント）による円滑なプログラミング演習の実施
  - 不揃いなPC＆ネット環境上に統一された利用環境を実現する。
  - 環境の利用について学生には何も「選ばせない」。
  - プログラミング以外の問題，PC＆ネット環境依存の問題で教員と学生を煩わせない。
- 学生の学修状況をオンライン／オンデマンド講義で俯瞰
  - クライアントからサーバに学生が今書いているコードを定期的にアップロード，一括表示し，遠隔では追いづらいクラス全体の学修を俯瞰できるようにする。
  - オンラインでもオンデマンドでも学生個人の学修の過程を把握し，適切な指導や助言を与えられるようにする。
  - 目の届かない遠隔プログラミング演習で学生がサボらないように指導する。

本取組みは対面授業でも有効であり，対面授業再開後もシステムは継続して利用されている。

本取組みの経緯

- コロナ禍以前
  - 第一著者担当の「オブジェクト指向プログラミング」で，Eclipseのプラグインで学生コードを回収，Web表示するシステムを構築していたが失敗。
  - 失敗の最大の原因はユーザインターフェースの使いにくさ。（学生をひとりひとりクリックしなければコードが見れず，クラス全体を俯瞰できなかった。）
- 2020年度前期（コロナ禍中）
  - 第一著者担当の「オブジェクト指向プログラミング」の開講前に本システムを開発し，運用に成功。
  - 開講に間に合わせ確実な運用を図る「欲張らない Minimalism の仕様と設計」方針が短期間での開発の成功に寄与。
  - インストーラ配付と学生の協力（バグレポート）が導入成功と安定運用に寄与。
  - cf.) Windows Subsystems for Linux（WSL）を使った他の遠隔演習では環境の整備や運用に関するトラブル対応に教員が忙殺。
- 2020年度後期～
  - 「オブジェクト指向プログラミング」の取組みを他科目に水平展開。
  - クライアントをフレームワーク化。
  - JABEE対応で科目毎／科目間で毎年度開講前後に教育改善の会議を実施する体制，教員間の連絡のよい学風が水平展開に寄与。
  - 開発と運用を並行実施する体制を確立。
  - システムの機能の追加と改善を継続。

本取組みの対象

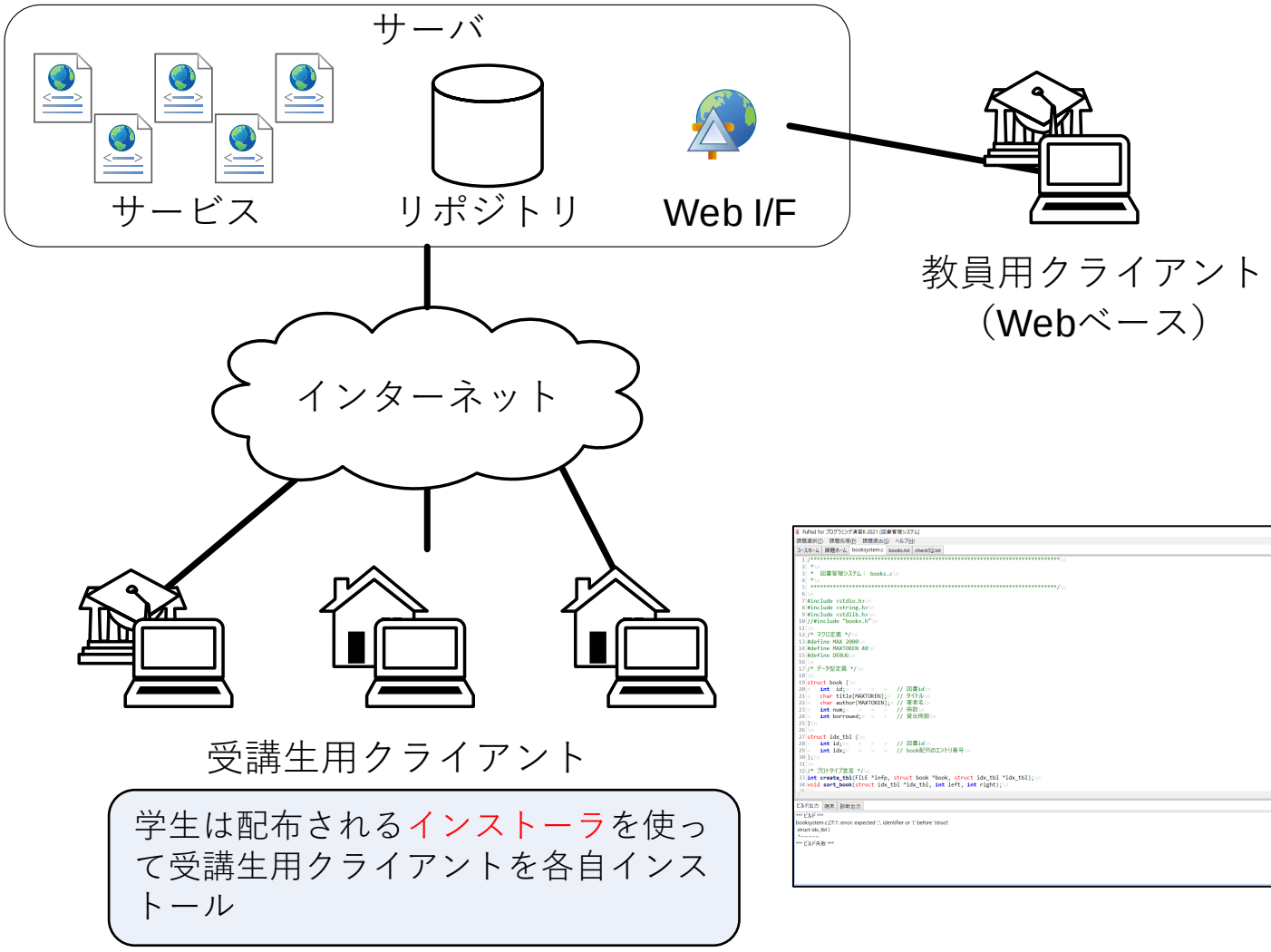
福岡大学工学部電子情報工学科に属する学部2～3年生

対象科目

科目	対象学年	開講期	言語	受講者数
オブジェクト指向プログラミング	3年次	2020年度前期	Java	87
プログラミング演習II	2年次	2020年度後期	C言語	74
オブジェクト指向プログラミング	3年次	2021年度前期	Java	74
プログラミング演習II	2年次	2021年度後期	C言語	74
電子情報工学実験	2年次	2021年度後期	アセンブリ言語（RISC-V）	185
計　3科目		計 4期		計　494

- オブジェクト指向プログラミングは教員2名＋TA1名で指導。
- プログラミング演習IIは教員3名＋TA7名で指導
- 電子情報工学実験は教員3名＋TA2名で指導
- 2022年度前期には計算機工学Iも導入予定

開発＆運用した遠隔演習環境の概要



学生は配布されるインストーラを使って受講生用クライアントを各自インストール

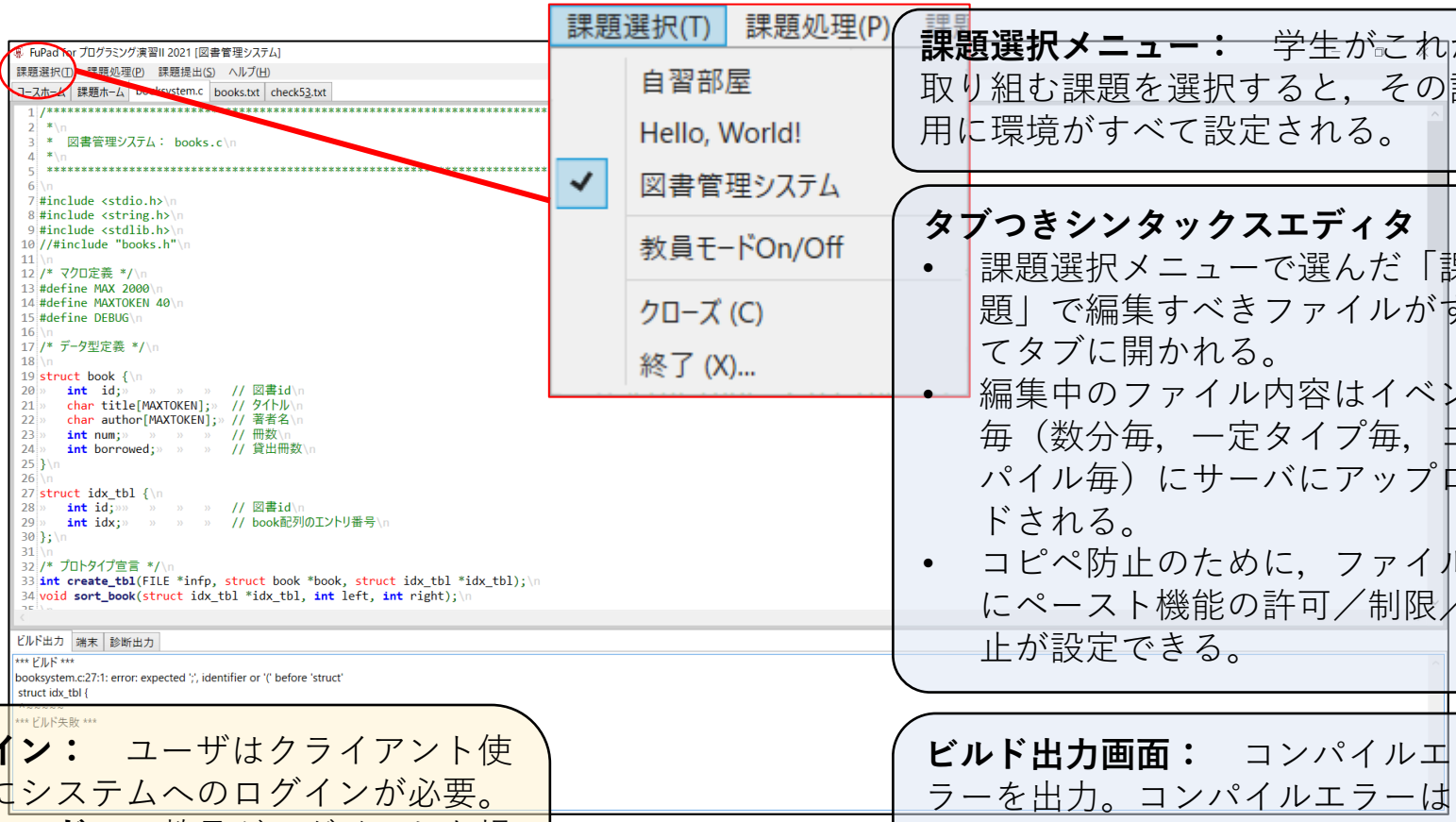
開発＆運用した遠隔演習環境のサーバ

- Apache（2021年度後期にNginxに移行済）で構築
- PHPで実装され，JSON，XMLでデータ交換する小さなサービス群を提供：ユーザ認証，アップロード，ダウンロード，チェックイン，チェックアウト，提出，マイルストーン管理（2021年後期廃止）
- 各科目の開講年度ごとにコースルートディレクトリを設け，コース記述ファイルとサービス群を配置
- 各科目の開講年度ごとに学生の成果物を蓄積するリポジトリを設置
- コース記述ファイル（XMLファイル）に，コースを構成する課題，各課題のファイル構成／提出／受講生用クライアントの環境設定／ビルド・実行プロセス等を定義，サービスとクライアントの振舞いを設定。

コース記述ファイルの記述例（一部）

```
<?xml version="1.0" ?>
<course>
  <task id="booksystem" enabled="true">
    <title>図書管理システム</title>
    <startupweb url="TaskHome_booksystem.php" />
    <file path="booksystem.c" paste="limited" highlighting="C++" />
    <file path="books.txt" paste="limited" />
    <file path="check5_3.txt" paste="limited" />
    <build>cmd /c "chcp 65001 > nul && gcc booksystem.c -o booksystem"</build>
    <execute cmdpos="0" cmdsurfix=".exe">chcp 65001 > nul && booksystem</execute>
    <submission id="step1.1">課題1.1: struct_ex.c のコンパイルと実行</submission>
    <submission id="step1.2">課題1.2: create_tbl 関数の作成</submission>
    ...
  </task>
  ...
</course>
```

開発＆運用した遠隔演習環境のクライアント（1）



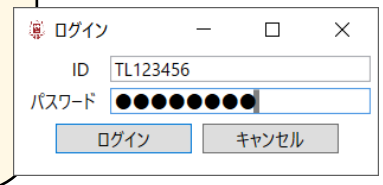
課題選択メニュー：　学生がこれから取り組む課題を選択すると，その課題用に環境がすべて設定される。

タブつきシンタックスエディタ

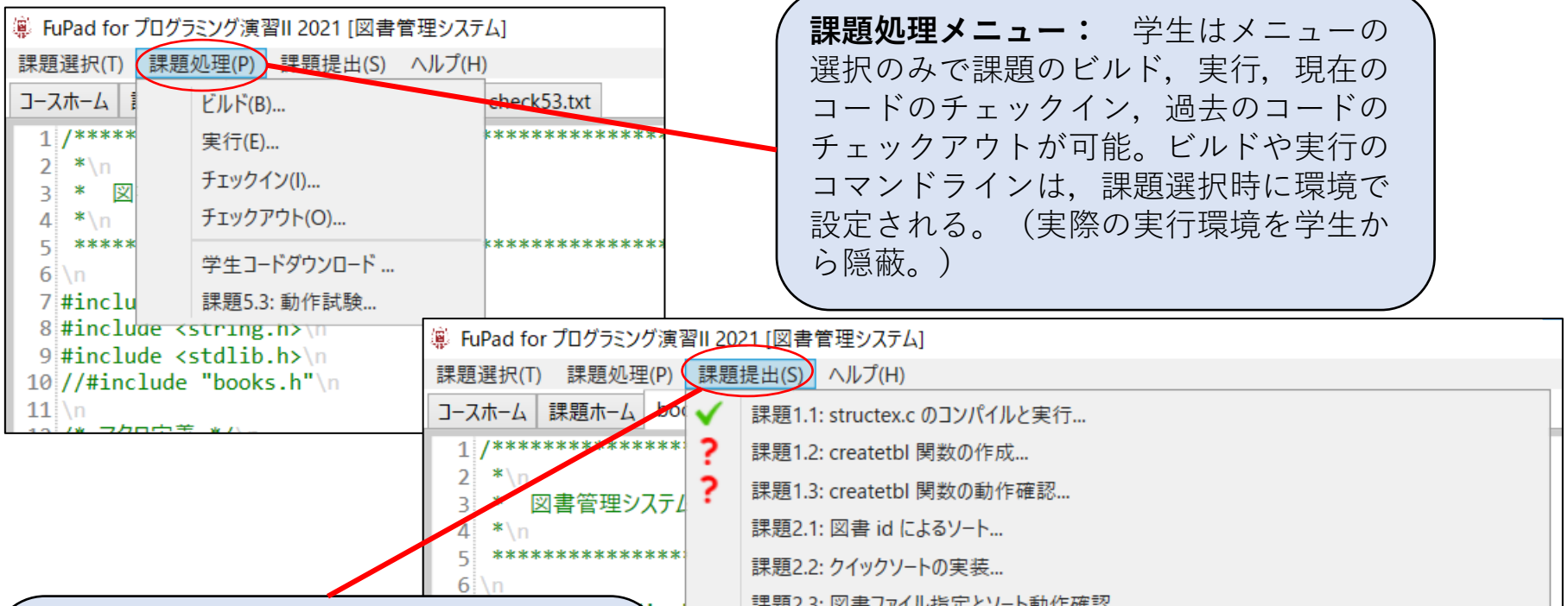
- 課題選択メニューで選んだ「課題」で編集すべきファイルがすべてタブに開かれる。
- 編集中のファイル内容はイベント毎（数分毎，一定タイプ毎，コンパイル毎）にサーバにアップロードされる。
- コピペ防止のために，ファイル毎にペースト機能の許可／制限／禁止が設定できる。

ビルド出力画面：　コンパイルエラーを出力。コンパイルエラーはサーバにアップロードされる。（その他に端末出力，診断出力画面も）

ログイン：　ユーザはクライアント使用前にシステムへのログインが必要。
教員モード：　教員がログインした場合は教員モードで動作。学生のコードをローカルにダウンロードして動作を確認できる。その他教員向けの機能が使用可能になる。



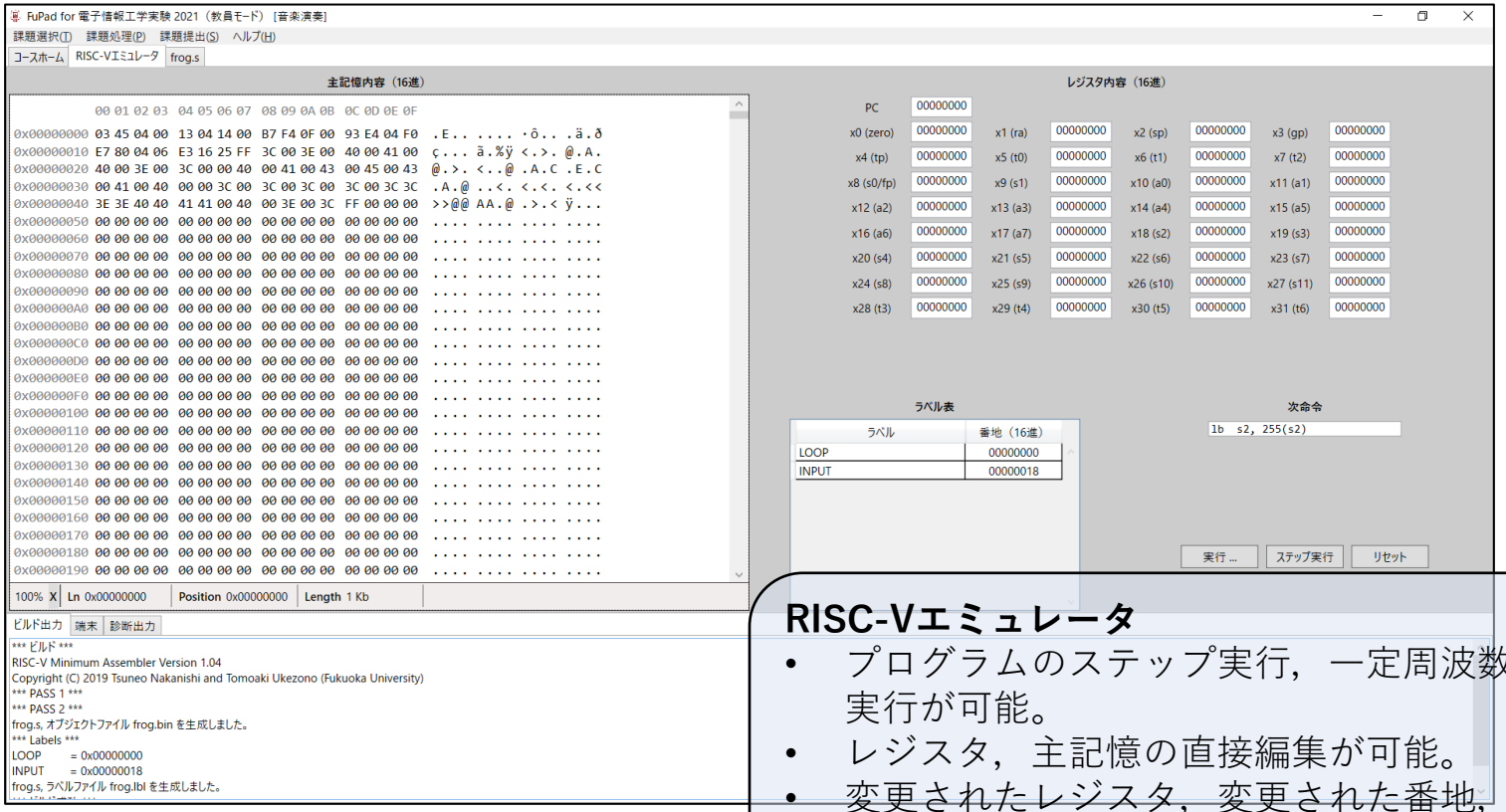
開発＆運用した遠隔演習環境のクライアント（2）



課題処理メニュー： 学生はメニューの選択のみで課題のビルド，実行，現在のコードのチェックイン，過去のコードのチェックアウトが可能。ビルドや実行のコマンドラインは，課題選択時に環境で設定される。（実際の実行環境を学生から隠蔽。）

課題提出メニュー： 学生はメニューの選択のみで，その時点のコードを課題を採点対象として提出可能。提出したコードが採点待ちの間は「❓」でマーク，教員のチェックに通ると「✓」でマーク。学生は採点の完了を待たずに次の課題に進んでよい。

開発＆運用した遠隔演習環境のクライアント（3）

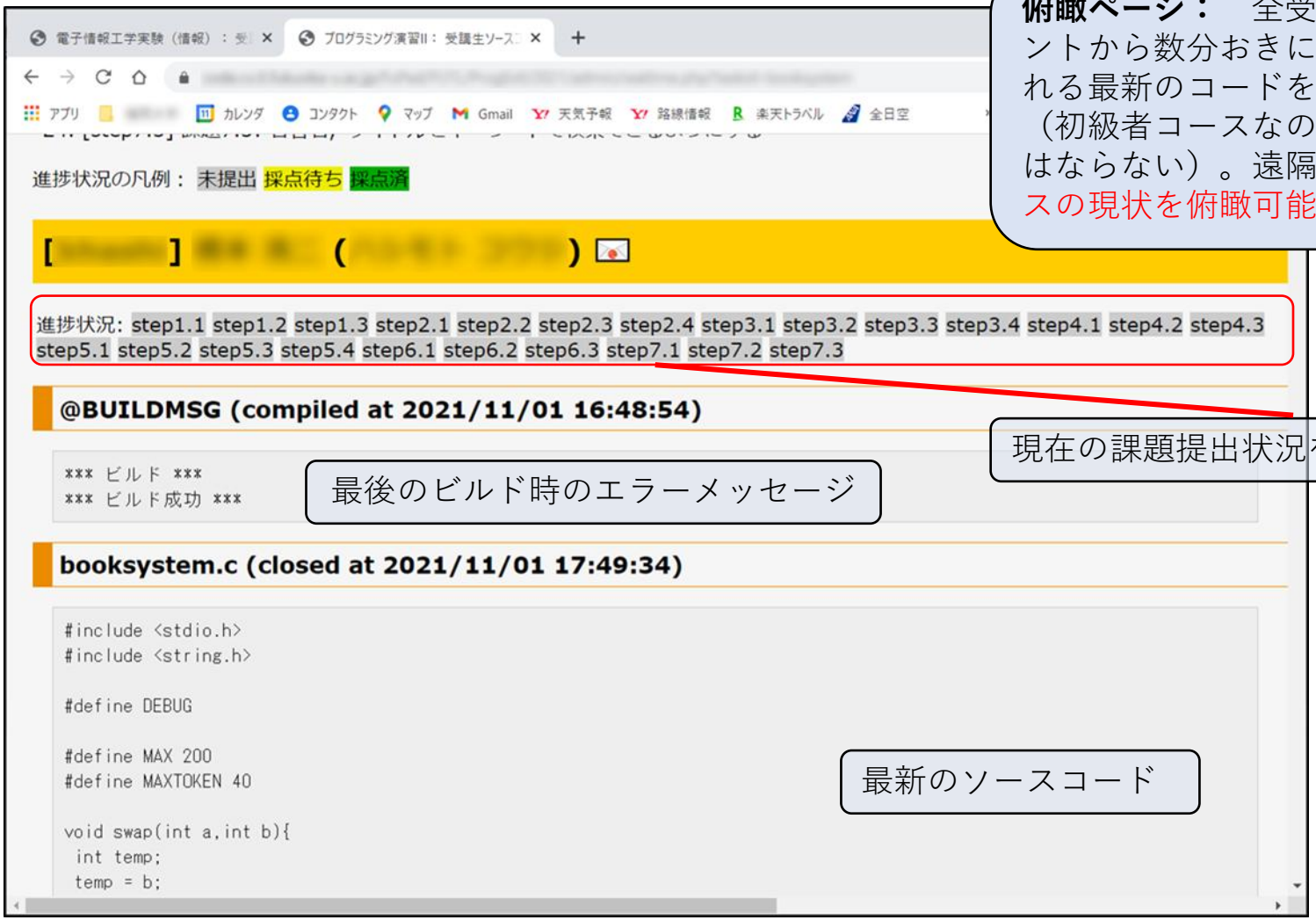


RISC-Vエミュレータ

- プログラムのステップ実行，一定周波数での実行が可能。
- レジスタ，主記憶の直接編集が可能。
- 変更されたレジスタ，変更された番地，次に実行する命令の番地のハイライトが可能。

電子情報工学実験版クライアント：通常のクライアントに加えて，RISC-Vのアセンブラ，プロセッサエミュレータ，独自ライブラリ（入出力，乱数，音楽演奏等）を統合し，アセンブリ言語・機械語の演習が可能。

開発＆運用した遠隔演習環境のWeb I/F（1）



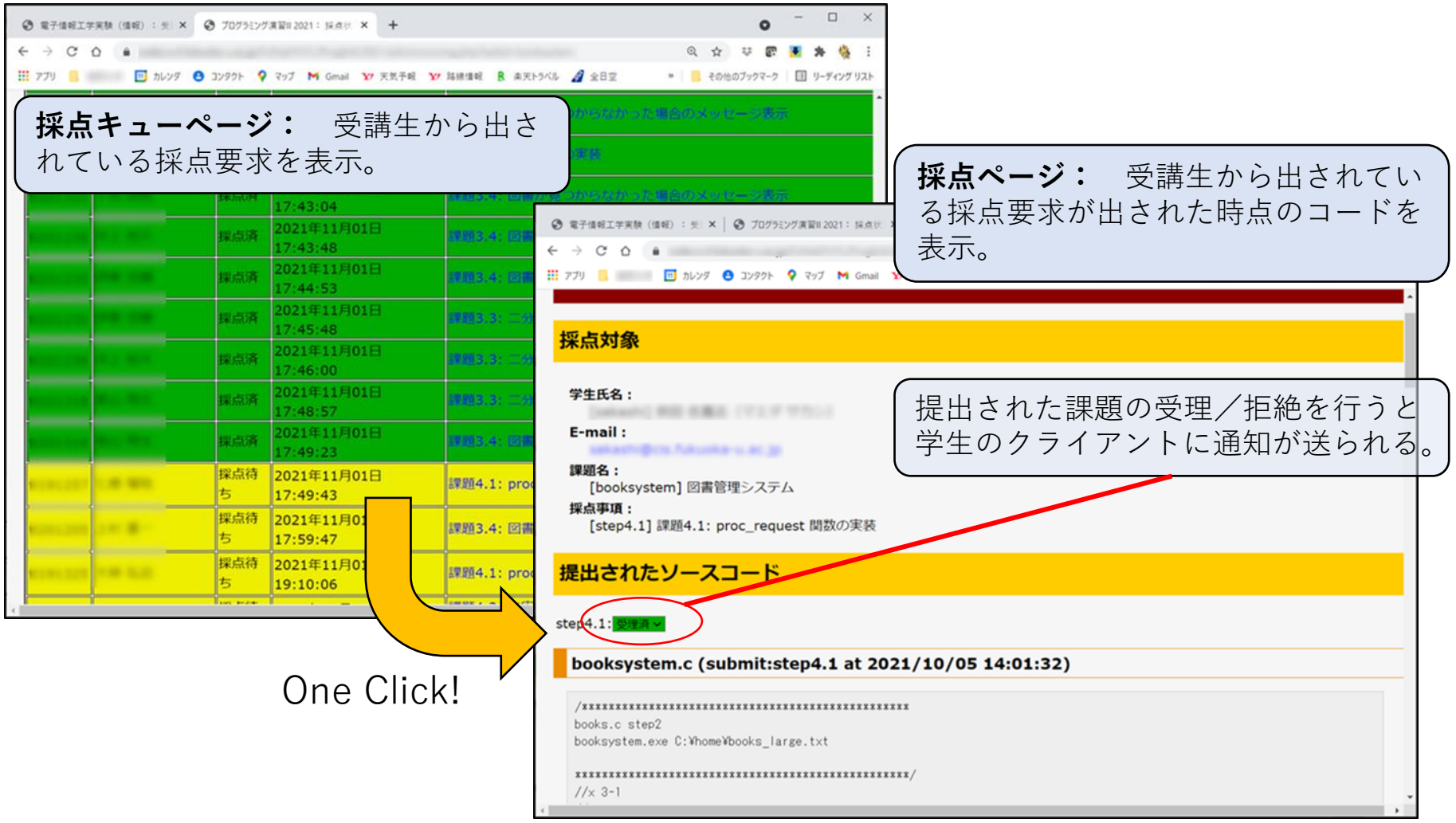
俯瞰ページ：　全受講生のクライアントから数分おきにアップロードされる最新のコードを1ページに表示（初級者コースなので大した規模にはならない）。遠隔であってもクラスの現状を俯瞰可能。

現在の課題提出状況をアイコン表示。

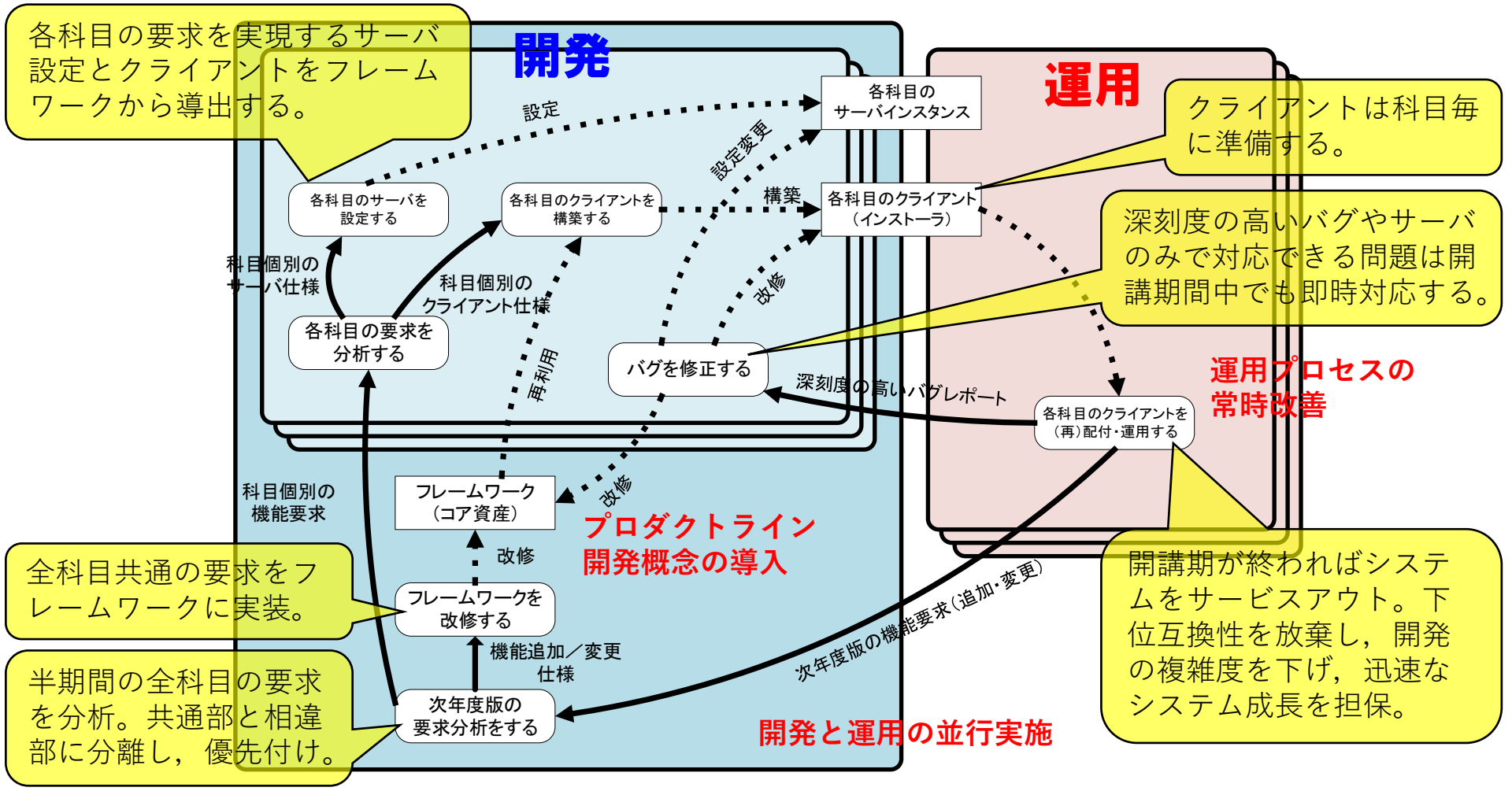
最後のビルド時のエラーメッセージ

最新のソースコード

開発＆運用した遠隔演習環境のWeb I/F（2）



遠隔演習環境の開発と運用のプロセス



開講期単位でのシステムと演習実施の継続的改善を図る。

オンライン／ハイブリッド演習での運用例

1. 教員は Microsoft Teams でその日の演習で学ぶ概念や演習の内容を解説する。
2. 学生は各自，受講生用クライアント上で課題選択し課題に取り組む。
3. 学生は課題が規定のところまで完成したら課題提出する。
4. 教員は演習中， Web I/Fの俯瞰ページで学生の演習の進捗状況を俯瞰し，ありがちなエラー等は Teams でクラス全体に注意する。
5. 教員はTAを監督し，学生の提出物のチェックを進める。
   - 教員／TAは提出されたコードをチェックし，課題の要求を充たしていたら受理，充たしていなければ拒絶する。
   - コードの審美性など高度な判断を伴う事項は教員が評価する。
   - TAは判断に迷うものは教員に対応を相談する。
6. 学生は質問があれば Teams 上で相談する。
   - 教員／TAと学生は，最新のコードをWeb I/Fで見るか，ローカルにダウンロードして相談に対応する。
7. 提出物のチェックや質問対応に関する業務割当は教員が Teams を用いて実施する。

システムとTeamsのチャット機能の併用でチームでの指導を実現

学生による本取組みの評価（概要）

2020年前期～2021年前期の3科目の受講生に任意のアンケート調査（受講生のべ235名中31名（10%超）が回答）

2021年度後期のプログラミング演習IIの受講生にアンケート調査（受講生61名（80%）が回答）

- ほぼすべての学生は特段のトラブルなく自力で受講生用クライアントをインストールできた。
- ほぼすべての学生は特段のトラブルなく自力でコードの編集，コンパイル，実行等の開発活動を実施できた。
- ほぼすべての学生は受講生用クライアントを用いたプログラミング演習に満足している。
- 受講生用クライアントによる演習と一般ツールによる演習について，半数の学生はどちらでもよいとし，残り半分の学生のおおよそ同数がどちらかを好んでいる。

学生による本取組みの評価（詳細）

- 当該遠隔演習環境による演習で満足している点
  - マイルストーン機能（「課題提出機能」の前身，2021年度後期に「課題提出機能」として再定義された）により提出した成果物が教員にチェックされたか否かがわかり，自身の進捗が確認できる。※
  - 遠隔講義にもかかわらず教員の指導を受けられる。
  - 開発環境の操作に惑わされることなく，コードを編集，コンパイル，実行できる。
  - Moodle やメールによる方法とはちがって，トラブルなく自動で成果物を提出できる。
- 当該遠隔演習環境による演習で不満な点
  - コピー＆ペースト機能が制限されるのはイライラする。
  - 提出したプログラムが本当に教員にチェックされたかわからない。※
  - 統合エディタの機能が貧弱である（自動フォーマット，コメントアウトができない。）
  - マイルストーン宣言した後にコードを編集してよいのかわからない，どの時点のコードが評価されているのかわからない。（2021年後期版の「課題提出機能」導入により改善済）

※で相反する評価がされているのは演習科目の実施体制が異なったため。教員とTAが多く，学生提出物の審査を短いターンアラウンドタイムでできた演習科目の受講生はスピーディなチェックに満足している。

教員による本取組みの振り返り

- 円滑化：　遠隔であっても円滑にプログラミング演習をできるようになった。（当初のねらい通り）
  - 学生がプログラミング以外のトラブルを抱えること，その対応に教員が時間をとられることがなくなった。
  - 学生の演習における操作環境が均一化され，PC環境のちがいの把握やそれによるトラブル対応に取られる時間が大幅に減った。
  - プログラミング上の直接的な問題への対応に時間が割けるようになった。
- 見える化：　完成時やビルド時のみならず，途中のコード編集の過程が追えるようになった。（当初のねらい通り）
  - 学生のコードがリアルタイムに俯瞰できるようになり，クラス全体に必要な指導や助言が把握できるようになった。（一部の学生の，同じような問題の対応に時間をとられすぎることがなくなった。）
  - 個々の学生がどのように考えてコードを書いたか思考の課程が追え，適切な助言や要重点説明箇所の把握ができるようになった。
  - いわゆる「コピペ提出」が見つけやすくなり，コピペ提出疑い者への提出物の説明要求のルール化もあって，従来見られたコピペ提出が大幅に減った。
  - フォローや指導が必要，かつ自ら質問できない学生の把握が容易になり，教員間でそうした学生に関する情報の提供が容易になった。（コードの時系列での伸び，クライアントの使用履歴，最終更新日時等で判断）
  - 但し，問題が見えるようになった結果，その対応に要する時間は増えた。

教員による本取組みの振り返り（つづき）

- 省力化：　システムによる教員業務の自動化により間接業務にかかる時間が削減された。
  - 不揃いな構成の成果物が提出されることがなくなり，採点時の成果物の把握や整理にかかる時間が削減された。
  - 成果物が揃った形式で蓄積された結果，採点支援のスクリプトを書いて一層の採点省力化を行うことが可能になった。
  - cf.) 以前は Moodle で提出された，不揃いな成果物を手作業でひとつひとつ採点していたため，手分けをしても相当な時間を要していた。
- コミュニケーションの改善：　システムによる情報共有とTeamsの併用により教員／TA／学生間のコミュニケーションが改善された。
  - 関係者が，同じ構成の成果物の同じスナップショットを，同じクライアント，あるいはWebで見られるようになり，学生のコードの状況把握や問題のある箇所をspecifyすることが容易になった。
  - 他のTAの指導が見えるようになり，TA同士で自身の指導判断が適切かクロスチェックできるようになった。
- 分業の効率化：　システムにより情報共有とプロセスの標準化がされた結果，教員とTAの分業が効率的にできるようになった。
  - 教員とTAの分業が容易になった。（教員はTAへの業務割当やTAの手に余る問題への対応，プロセス通りにできるものや簡単な指導はTA）
  - TAへの業務割当を動的，かつ全体的に最適化できるようになった。（Teams併用の効果）
  - 能力に応じてTAを適材適所に割り当てられるようになった。（プロセス通りの形式的なチェックをする役，コード改善など一部高度な判断を要する指導をする役）
  - 従来のように遊んでいるTAが居なくなり，TAひとりで処理できた業務量も増えた。

本取組みで得られた知見と教訓

- リテラシや本質外の問題（全角スペース，スペルミス，セミコロン忘れ）でビルドまでに躓いている学生は少なくない。
- 多くの学生はプログラミング演習における教員の「寄り添い」を欲している。
- 最終成果物から分かることは少なく，コード編集の過程にこそプログラミング教育改善のための情報が埋まっている。
- 学生の学修の姿勢や進捗にはパターンがある。
- システムのユーザインターフェースは（論文では語られにくいものの）取組みの成否に支配的な影響をもたらす。
- 「見える化」は見えていなかった問題，見なかったことにしていた問題を顕在化し教員の仕事量を増やし得る。
  - しかし，これらは本来取り組むべき本質的な問題である。
  - 省力化と効率化で仕事量を削減し，その「稼ぎ」を顕在化された問題への対応，その一般的解法の確立にあてるべきである。
- 大学教育DXは教員間の意思疎通，協働体制と協働プロセスの確立が要諦となる。

システムの将来性

- コンピュータと人の分業の洗練化
  - コンピュータによるソフトウェアテストやソースコード静的解析により採点・指導を一部自動化することで教員の負担を軽減。
  - ルールベースの処理でできないものの，パターン認識が利く診断はいわゆる人工知能（機械学習）に任せ，さらに教員の負担を軽減。（そのためのデータの収集と蓄積，学習モデル構築を継続。）
  - 教員はアーキテクチャやコードの審美性など人間しかできないことを担当。
- プログラミング教育の人材・空間・時間制約の緩和
  - キャンパスや教室をまたいだプログラミング教育
  - 大学から高校・中学・小学へのデジタル出張講義
  - 過疎地・山村・離島の学校での遠隔プログラミング教育（プログラミング教育デバイドの解消）
  - 企業における本社と支社，オフィス勤務者と在宅勤務者をまたいだ研修
  - プログラミング教育提供元での専門人材の集積とノウハウ蓄積
  - 隙間時間で働く在宅非常勤講師の活用（人材確保の効率化）
- プログラミング教育のアウトソーシングの実現
  - プログラミング教育専門企業からその顧客企業や教育機関への演習提供
  - 人件費の安価な国でのプログラミング指導教員の雇用