



Education System Based on Algorithm (ESBOA): Design and Simulation-Based Evaluation of an Algorithmic Framework for Teaching, Learning, Assessment, and Educational Management

Sayed Mohammad Kazim Rajaei^{1*}, Musawer Hakimi²

¹ Computer Science, Bakhtar University, Kabul, Afghanistan.

² Computer Science Department, Samangan University, Samangan, Afghanistan.

Received: July 10, 2026

Revised: August 20, 2026

Accepted: October 12, 2026

Published: October 31, 2026

Corresponding Author:

Sayed Mohammad Kazim Rajaei

kazimrajei4@gmail.com

DOI: [10.29303/jppipa.v12i10.16401](https://doi.org/10.29303/jppipa.v12i10.16401)

 Open Access

© 2026 The Authors. This article is distributed under a (CC-BY License)



Abstract: Universities have teaching, assessment, feedback, and management, yet these parts are rarely joined into one explicit and repeatable process. This paper presents the Education System Based on Algorithm (ESBOA), a framework that treats the educational lifecycle as an algorithm: Input, Objective, Instructions, Teaching, Learning, Assessment, Feedback, and Output, with planning, monitoring, reporting, and improvement running through every stage. The study has three distinct parts. First, framework development: using a design-science approach, recurring problems of traditional teaching were traced to six governing principles, an eight-stage architecture, an input-process-output model, and pseudocode for a bounded corrective-learning loop with teacher referral. Its contribution is not a new learning theory but an explicit, auditable procedure that joins objective-level assessment, a capped feedback cycle, and management reporting, which mastery learning, outcome-based education, adaptive systems, and learning analytics each address only in part. Second, computational verification: because the planned pilot has not yet taken place, the loop was verified in a Monte Carlo simulation of 200 synthetic students over 500 replications, followed by sensitivity analyses of feedback gain, ability, difficulty, cycle cap, assessment error, forgetting, and objective dependency. Under assumed parameters, simulated mean mastery was 0.57 in a single-pass course, 0.83 with the feedback loop, and 0.86 with student-level analysis added; these values follow from the model's assumptions and are not independent evidence of effectiveness. The benefit of the loop disappeared when feedback carried no information, and the loop required about 2.3 to 2.4 corrective cycles per student. Third, educational effectiveness has not been tested. A twelve-month pilot roadmap, evaluation indicators, a risk register, and ethical safeguards are proposed for computer science courses. The findings support the conceptual and computational feasibility of ESBOA only; whether it improves learning must be established empirically.

Keywords: Algorithmic education; Computer science education; Design-science research; Educational management; Feedback loop; Monte Carlo simulation; Objective-based assessment

Introduction

Think of a typical university course. The syllabus lists objectives, the lecturer teaches in the way that has worked before, the exam covers what is easy to mark,

and the grades arrive after the semester is over. Every step is reasonable on its own. The trouble lies in the links between them: objectives do not always shape the teaching, the exam does not always test the objectives, and the results rarely change what happens next. Biggs

How to Cite:

Rajaei, S. M. K., & Hakimi, M. (2026). Education System Based on Algorithm (ESBOA): Design and Simulation-Based Evaluation of an Algorithmic Framework for Teaching, Learning, Assessment, and Educational Management. *Jurnal Penelitian Pendidikan IPA*, 12(10), 255–266. <https://doi.org/10.29303/jppipa.v12i10.16401>

(1996) described the remedy as constructive alignment, in which objectives, teaching activities, and assessment are designed together (Biggs & Tang, 2011). Courses designed this way have been linked to deeper student approaches to learning (Wang et al., 2013), and outcome-based education applies a related logic at programme level (Harden, 2007). Alignment is easy to endorse and much harder to maintain week after week, across many instructors, with limited time.

Research on feedback shows both the promise and the difficulty. Formative assessment can raise achievement (Black & Wiliam, 1998), and feedback works best when it tells learners where they are going, how they are doing, and what to do next (Hattie & Timperley, 2007). A meta-analysis of 435 studies found a medium average effect ($d = 0.48$) but wide variation, with the content of the feedback shaping how much it helped (Wisniewski et al., 2020). Sadler (1989) explained why: feedback closes a gap only when learners know the standard, see how their work compares with it, and act to narrow the difference. Mastery learning builds this cycle into teaching by pairing short units with formative checks and corrective work (Guskey, 2007). Its record is mixed. One meta-analysis reported positive effects on examinations (Kulik et al., 1990), while a best-evidence review urged caution about gains on standardized tests (Slavin, 1987). A recent practical review reports moderate effect sizes and recommends cumulative testing to check retention (Winget & Persky, 2022).

Technology adds options but not always a whole-process view. A review of 146 studies grouped artificial intelligence (AI) applications in higher education into profiling and prediction, assessment and evaluation, adaptive systems and personalisation, and intelligent tutoring, and it noted weak links to pedagogical theory and little reflection on risks (Zawacki-Richter et al., 2019). A later review of reviews found adaptive systems and personalisation especially prominent and called for more attention to ethics and rigour (Bond et al., 2024). A systematic mapping of AI-enabled adaptive learning systems likewise found few systems implemented in real teaching settings (Kabudi et al., 2021). Evidence for individual tools is encouraging. Intelligent tutoring systems have shown positive effects (Kulik & Fletcher, 2016), and learning analytics have supported study success, although success may not come from analytics alone (Ifenthaler & Yau, 2020; Azevedo et al., 2021). What is harder to find is a plain process model that joins objectives, instruction, assessment, feedback, and management reporting in one repeatable loop that teachers can inspect and overrule.

These strands each address one part of the problem. Mastery learning supplies the corrective cycle but is usually designed at the level of a course unit (Guskey, 2007; Kulik et al., 1990; Winget & Persky, 2022). Outcome-based education and constructive alignment supply the principle that outcomes, teaching, and assessment must agree (Biggs & Tang, 2011; Harden, 2007) but say little about the operational rule that turns an assessment result into a next step. Adaptive learning systems and intelligent tutors personalise within a tool (Kabudi et al., 2021; Kulik & Fletcher, 2016), and learning analytics supplies monitoring and prediction (Ifenthaler & Yau, 2020), yet neither is by itself a course-level process that a department can inspect, audit, and overrule. The gap this paper addresses is therefore narrower than the absence of any one of these ideas: it is the absence of one explicit procedure that connects them, from objectives to management reporting. Table 5 in the Results and Discussion section compares ESBOA with each approach in turn.

Computer science supplies a vocabulary for such a model. An algorithm is a defined sequence of steps that turns inputs into an expected output, and thinking in these terms is a core part of computational thinking (Wing, 2006). The Education System Based on Algorithm (ESBOA) applies that idea to education. Its guiding sentence is simple: from objectives to outcomes through algorithmic steps. ESBOA does not propose to replace teachers with software. It proposes to write the educational process down as explicit steps and decision points, so that it can be taught, checked, and improved.

The aim of this paper is to present ESBOA as a design and to examine its internal logic before it is tried with real students. Four research questions guide the work. They condense the ten sub-questions of the original ESBOA proposal.

RQ1. Which components, inputs, and rules does an algorithmic education process need, and how can objectives be turned into explicit procedures?

RQ2. How can student-level analysis, objective-based assessment, and feedback be linked so that a learning gap triggers corrective learning?

RQ3. What outputs and management information can the framework generate?

RQ4. How feasible is ESBOA for computer science courses, and what should a pilot measure?

The paper makes four contributions: a specified architecture with six principles, a bounded corrective-learning algorithm, a simulation that verifies the loop and shows when it helps and when it does not, and a pilot and evaluation plan. Their status differs: the first two are design outputs, the third is computational

verification, and the fourth is a plan. None of them is evidence that ESBOA improves learning.

Most components of ESBOA are not new, and the paper does not claim that they are. Objective-based assessment, corrective feedback, mastery thresholds, and monitoring are established in the literature cited above. The claimed contribution is of a different kind. In the terms of Gregor et al. (2013), ESBOA is an improvement-level design contribution rather than a new learning theory. It offers (a) a stage architecture that joins instructional and management processes in one lifecycle; (b) a corrective-learning algorithm bounded by a cycle cap R and a teacher-referral rule, which separates automated routing from professional judgement; (c) a traceability matrix from recurring problems to principles to mechanisms, so that each element can be audited; and (d) a simulation protocol that keeps algorithm verification apart from educational validation. The practical value expected from these elements lies in educational quality (every assessment item tied to an objective), instructional consistency (a shared instruction plan across instructors), and institutional decision-making (objective-level records, referral counts, and cycle counts that give managers evidence). These are propositions for the pilot to test, not findings.

Although the first implementation targets university computer science, the framework is relevant to science education more broadly. Computer science is a STEM discipline with procedural, objective-level outcomes, such as tracing a program or designing a query. Laboratory and conceptual science courses share the same structure of measurable objectives, practice, and corrective feedback, and computer simulation is itself an established tool in science education research (Rutten et al., 2012). The stages and the loop are discipline-neutral, but this paper offers no evidence for science courses outside computer science, and transfer is treated as a hypothesis for later studies.

Method

Research Design

This is a design-science study in the sense of Hevner et al. (2004) and Peffers et al. (2007): a problem is identified, a solution is designed, and the design is demonstrated and then evaluated. The paper covers problem identification, objectives, design, and a demonstration by simulation. In the vocabulary of Venable et al. (2016), the evaluation reported here is artificial and formative: it checks the design in a controlled model before any naturalistic, summative evaluation. Evaluation with real students in a university computer science department is planned but

has not been carried out. For that reason, no claim in this paper rests on classroom data. Two kinds of checking are kept apart throughout. Verification asks whether the algorithm and its code behave as the design says. Validation asks whether a model represents real learning well enough for its intended use (Sargent, 2013). This paper reports verification only.

Framework Development

The starting point was a research proposal for a pilot in a university computer science department. Its list of ten recurring problems in traditional teaching, such as unclear objectives, assessment that does not match objectives, and little use of data in management, served as the requirements list. The derivation then followed four steps. First, each problem was restated as a requirement. Second, each requirement was matched to a design principle and, where one was needed, a working mechanism; this produced the six principles and the traceability matrix in Table 3. Third, the principles were ordered by the logic of an input-process-output system, which gave the eight-stage lifecycle with a management layer. Fourth, the feedback and mastery elements were checked against the literature reviewed in the Introduction. The result is described in the Results and Discussion section and summarised in Figure 1 and Tables 2 and 3. This derivation is a documented design rationale, not a formal requirements study. The problem list comes from the authors' proposal rather than from a systematic survey, and the principles and stages have not yet been reviewed by external experts or evaluated against structured design criteria. These are limitations (see Limitations and implications). Expert review of the problems, principles, and stages by computer science teachers and quality-assurance staff is therefore planned for the framework-refinement phase of the pilot roadmap.

Simulation Model

To check that the corrective loop behaves sensibly, a Monte Carlo simulation was written in Python. Each synthetic student has a latent ability θ drawn from a standard normal distribution. A course unit has $J = 6$ objectives of increasing difficulty b_j . After c corrective cycles, student i has mastered objective j with the probability in Equation 1, where a is discrimination, g is the gain from instruction, and δ is the gain contributed by each cycle of feedback and correction. At $c = 0$ this describes a single teaching pass.

$$P_{ij}(c) = \frac{1}{1 + e^{-a(\theta_i + g + c\delta - b_j)}} \quad (1)$$

Only objectives not yet mastered are retaken in a cycle, and the chance of conversion in each cycle is set

so that the cumulative probability of mastery equals Equation 1. This avoids a common shortcut in which repeated attempts raise scores by luck alone: when $\delta = 0$, extra cycles add nothing. A student's mastery index is the share of objectives mastered (Equation 2).

$$M_i = \frac{1}{J} \sum_{j=1}^J m_{ij} \quad (2)$$

Three conditions were compared. In the conventional condition, instruction is delivered once and assessed once. In the ESBOA feedback-loop condition, up to $R = 3$ corrective cycles follow the first assessment. In the ESBOA full condition, student-level analysis adds a small level-matching gain μ to instruction before the first pass, and the loop follows. Table 1 lists all values. Each condition was run for 500 replications of 200 students, and the intervals reported below are the 2.5th to 97.5th percentiles across replications. The complete Python script, with fixed random seeds (base seed 62000; replication r uses seed $62000 + r$), is provided as Supplementary File S1 and will be deposited in a public repository on acceptance, so that every table and figure of the simulation can be regenerated.

Table 1. Simulation Settings

Setting	Symbol	Value
Students per run	N	200
Objectives per unit	J	6
Student ability	θ	Normal(0, 1)
Objective difficulty	b_j	-1.0 to 1.0, equal steps
Discrimination	a	1.2
Baseline instruction gain	g	0.3
Level-matching gain (full)	μ	0.2
Feedback gain per cycle	δ	0.5
Maximum corrective cycles	R	3
Mastery threshold	τ	0.8
Replications	-	500

The values in Table 1 were chosen as follows, and none is an empirical estimate. Ability follows the usual standard-normal convention for latent-trait models. Difficulties are spread evenly across the central two standard deviations of ability, so that a unit contains both easy and hard objectives. A discrimination of 1.2 is a moderate slope. The baseline gain g was set so that a single teaching pass leaves a little over half of the objectives mastered, which leaves room for correction to matter. The feedback gain $\delta = 0.50$ is half a standard deviation of ability per cycle, similar in size to the average feedback effect reported by Wisniewski et al. (2020) ($d = 0.48$); because it accumulates over up to three cycles, it is optimistic, which is why δ is varied from 0 to 0.75 below. The level-matching gain $\mu = 0.20$ is deliberately smaller than δ . The cap $R = 3$ and the

threshold $\tau = 0.80$ are design choices. Because mastery is the share of six objectives, a threshold of 0.80 means at least five of six objectives.

To test whether the conclusions hold under other assumptions, the three conditions were re-run (500 replications each, same seeds) with one change at a time: feedback gains of $\delta = 0, 0.25$, and 0.75 ; baseline gain $g = 0$; lower discrimination ($a = 0.8$); a lower mean ability (-0.5) and a wider spread of ability (SD 1.5); a wider difficulty range (-1.5 to 1.5); cycle caps of $R = 2$ and $R = 5$; and a mastery threshold of $\tau = 0.90$. Four extensions relax simplifications of the base model. Heterogeneous feedback lets δ vary between students (normal, mean 0.50, SD 0.25). Assessment error misclassifies each objective-level result with probability 0.10 in both directions, so that some mastered objectives trigger an extra cycle and some unmastered ones are accepted. Forgetting removes each mastered objective with probability 0.20 before a delayed check. Prerequisite dependency makes mastery of an objective fail with probability 0.30 when the preceding objective is not mastered. These extensions are simple stand-ins for real processes, not calibrated models.

The simulation can answer questions about the logic of the framework: does the loop help, for whom, and how does the benefit depend on the quality of feedback? It cannot say how much real students would gain. Because δ and μ are positive inputs, improved mastery is partly built into the model. The simulation therefore verifies that the algorithm and code behave as designed; it does not validate ESBOA as an educational intervention. Validation needs real data, which is the purpose of the pilot.

Planned Pilot Evaluation

The planned pilot uses a mixed-methods design. Quantitative measures include pre-test and post-test results, achievement of objectives, assignment and project performance, progression, participation, and time spent on selected activities. Qualitative sources include student questionnaires, teacher and management interviews, classroom observation, and records of implementation problems. Where it is institutionally and ethically feasible, ESBOA-supported courses will be compared with baseline or comparison-course data using a standardised mean difference (Equation 3). The population, sample, indicators, and statistical methods will be fixed before implementation.

$$d = \frac{\bar{x}_E - \bar{x}_C}{s_p} \quad (3)$$

Here $\bar{x}_E$ and $\bar{x}_C$ are the means of the ESBOA and comparison groups, and s_p is the pooled standard deviation. The pilot will proceed only after institutional

approval, with informed consent where required and the privacy protections described later.

Result and Discussion

Three kinds of material follow, and each is marked as such. The architecture, principles, and decision logic (Figures 1-3 and Tables 2-5) are design outputs. The simulation results (Figures 4-5 and Tables 6-7) are computational verification outputs. The pilot roadmap, indicators, and risk register (Figure 6 and Tables 8-9) are proposals that have not been implemented. None of them is evidence that ESBOA improves learning, teaching quality, or educational management.

Architecture of ESBOA

ESBOA arranges education as eight linked stages (Figure 1): Input, Objective, Algorithm/Instructions, Teaching, Learning, Assessment, Feedback, and Output. Educational management is not a ninth stage but a layer across all of them, carrying planning, monitoring, reporting, and improvement. The loop from Feedback back to Learning is the heart of the design. If assessment shows that an objective has not

been met, the process does not move on; it returns to corrective learning.

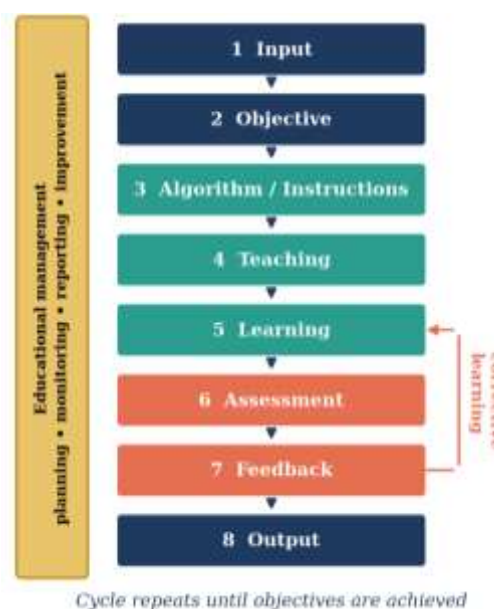


Figure 1. The ESBOA lifecycle. Stages 6 and 7 feed a corrective loop back to learning, and educational management runs across all stages

Table 2. ESBOA Stages, Functions, and Typical Records

Stage	Main function	Typical record
Input	Collect student, curriculum, resource, and teacher information	Student and course data
Objective	Define goals, competencies, and standards	Measurable objective list
Algorithm / Instructions	Set rules, procedures, and decision logic	Instruction plan
Teaching	Apply methods, materials, and activities	Session plans
Learning	Study, practice, and collaborate	Activity log
Assessment	Measure learning with tests, projects, and evaluation	Objective-level scores
Feedback	Analyse results and guide correction	Feedback record
Output	Generate achievements, profiles, and reports	Reports

Table 2 states the function of each stage and the record it should leave behind. These records matter, because they let a department see, and later audit, what actually happened in a course.

The stages are deliberately plain. Nothing in them depends on a particular platform, and a paper-based course could follow them. Technology enters only where it helps, for example to store records, to analyse student level, or to suggest corrective tasks.

Principles and Traceability

Six principles govern the stages: clearly defined objectives, explicit instructions, sequential activities, objective-based assessment, feedback and correction, and measurable output. Table 3 shows how they answer the problems that motivated the framework. The traceability is the point. Each problem is met by at least one principle and one working mechanism, so a reviewer can ask of any ESBOA element which problem it is there to solve.

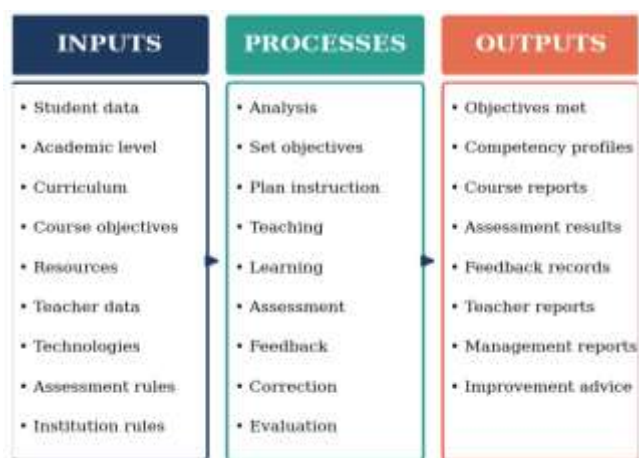
Table 3. From Problems to Principles and Mechanisms

Problem in traditional practice	Principle	ESBOA mechanism
Unclear or inconsistent objectives	P1 Clear objectives	Objectives written and made measurable before teaching
Teaching and outcomes drift apart; procedures vary by instructor	P2 Explicit instructions; P3 Sequential activities	Shared instruction plan with ordered activities
Assessment does not match objectives	P4 Objective-based assessment	Each assessment item mapped to an objective
Little systematic feedback	P5 Feedback and correction	A gap triggers feedback and a corrective task
Hard to produce comparable results	P6 Measurable output	Competency profiles and assessment records
Weak monitoring; little data in management	P6 plus management layer	Planning, monitoring, reporting, improvement
Limited personalization	Level analysis (stage 3 rule)	Student level selects teaching methods

Two cautions apply. First, principles are only as good as the objectives and instruments behind them: poorly written objectives will make an orderly process produce poor results. Second, the framework relies on feedback that is specific and usable. This echoes Shute (2008) on formative feedback and Nicol et al. (2006), who argue that feedback should support learners' own regulation and dialogue instead of only delivering verdicts.

Input-process-output view

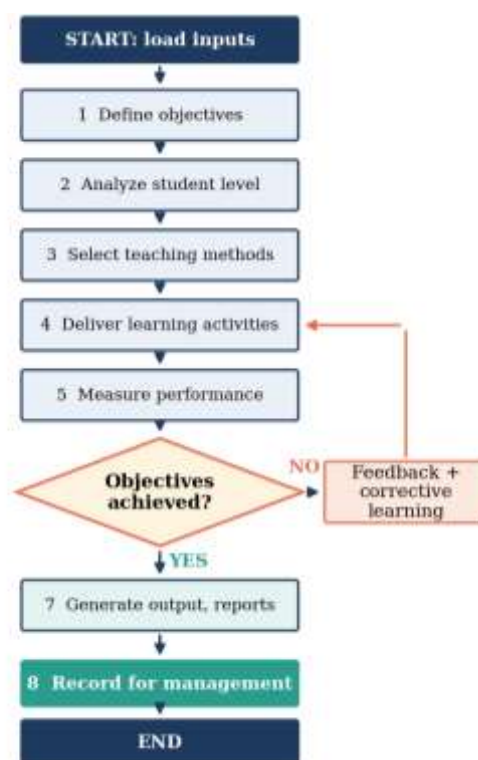
Seen from outside, ESBOA is an input-process-output system (Figure 2). Inputs include student information and academic level, curriculum, course objectives, resources, teacher information, available technology, assessment requirements, and institutional standards. The processes run from analysis and objective setting through teaching, learning, assessment, feedback, and correction to evaluation. Outputs include achieved objectives, competency profiles, course and teacher reports, assessment results, feedback records, management reports, and improvement recommendations. Framing the system this way helps a department decide what it must collect before the first class and what it should be able to report after the last.

**Figure 2.** ESBOA as an input-process-output model

Algorithm and Decision Logic

The decision logic appears in Figure 3 and as pseudocode in Algorithm 1. The original proposal let the loop return to learning activities until objectives were achieved. Two refinements are made here. A cap of R corrective cycles prevents an endless loop, and when the cap is reached the case goes to the teacher for review instead of being settled automatically.

The cap and the review step are design choices with a purpose. Without a cap, the workload of repeated assessment can grow quickly. Without teacher review, the system would treat its own threshold as unquestionable, which the proposal explicitly warns against. Who sets the threshold τ for “achieved”, and whether it differs by objective, is a decision for each department and should be written into the instruction plan.

**Figure 3.** Flowchart of the ESBOA core loop with the corrective-learning branch

```

INPUT : student_data, curriculum,
        resources, teacher_info
OUTPUT: learning_outcome, assessment_report,
        competency_profile
1 objectives <- DEFINE(curriculum)
2 level <- ANALYZE(student_data)
3 methods <- SELECT(objectives, level,
                    resources)
4 c <- 0
5 REPEAT
6   DELIVER(methods)
7   scores <- MEASURE(objectives)
8   gap <- COMPARE(scores, objectives)
9   IF gap is empty THEN EXIT LOOP
10  feedback <- FEEDBACK(gap); c <- c + 1
11  methods <- ADJUST(methods, feedback)
12 UNTIL gap is empty OR c = R
13 IF gap is not empty THEN REFER_TO_TEACHER(gap)
14 output <- GENERATE(scores, level)
15 RECORD(output) FOR management

```

Figure 4. Algorithm 1: ESBOA core loop (pseudocode)

What ESBOA adds

ESBOA is not a rival to learning management systems, adaptive learning, or learning analytics, and it does not claim that these tools are missing. Its claim is narrower: an overarching process model that ties the pieces together. Table 4 compares it conceptually with conventional practice and with the single-purpose tools identified by Zawacki-Richter et al. (2019), and Table 5 positions it against the four established approaches discussed in the Introduction. Both comparisons describe design intent. They are not measured results.

Table 4. Conceptual Comparison of Approaches

Feature	Conventional practice	Single-purpose tools	ESBOA
Scope	Course-by-course routines	One slice, such as prediction, tutoring, or grading	Whole lifecycle, input to output
Objectives	Often implicit	Varies by tool	Explicit and measurable at the start of each cycle
Assessment link	Varies	Tool-specific	Every item mapped to an objective
Feedback to action	Optional, often late	Hints or alerts	Decision rule with a bounded corrective loop
Management	Separate reporting	Dashboards	Built-in planning, monitoring, reporting, improvement
Human role	Teacher decides	Mixed	Teacher oversight of automated decisions

Table 5. Positioning of ESBOA Relative to Established Approaches

Approach	Established contribution	What ESBOA adds
Mastery learning (Guskey, 2007; Winget & Persky, 2022)	Short units, formative checks, and corrective work until a threshold is met	Course-level lifecycle; cycle cap R with teacher referral; records for management
Outcome-based education and constructive alignment (Biggs & Tang, 2011; Harden, 2007)	Outcomes, teaching, and assessment designed to agree	An operational rule that turns an assessment gap into a corrective step; stage records for audit
Adaptive learning and intelligent tutoring (Kabudi et al., 2021; Kulik & Fletcher, 2016)	Personalised paths and hints inside a tool	A tool-independent process that a teacher can inspect and overrule; student level informs method choice
Learning analytics (Ifenthaler & Yau, 2020; Slade & Prinsloo, 2013)	Monitoring, prediction, and dashboards	Analysis tied to a defined decision (feedback, correction, referral), with ethical safeguards inside the process

Table 5 keeps the core idea of each established approach and names only what ESBOA adds to it. The differences are of integration and operation, not of learning theory.

If ESBOA proves useful, it will probably be because of this joining-up rather than any single component. Its novelty is therefore relative: ESBOA is an integrating design, and its distinctive elements are the decision logic with a bounded loop and teacher referral, the traceability matrix, and the management layer. The reviews cited earlier also indicate where to be careful: ethics and rigour (Bond et al., 2024), pedagogical grounding (Zawacki-Richter et al., 2019),

and the dilemmas of using student data (Slade & Prinsloo, 2013; Holmes et al., 2022).

Simulation Results

Table 6 and Figure 5 summarize the simulation. In the conventional condition, students mastered on average 0.57 of the objectives, and 0.29 of students reached the 80% threshold. Adding the feedback loop raised mean mastery to 0.83 and the share at or above the threshold to 0.73. Adding student-level analysis raised them further to 0.86 and 0.78. These differences are produced by the assumed feedback and level-matching gains. They show that the algorithm and code behave consistently with the model's assumptions, not that real students would improve.

Table 6. Simulation Results (500 Replications of 200 Synthetic Students)

Measure	Conventional	ESBOA loop	ESBOA full
Mean mastery	0.57	0.83	0.86
95% interval	0.53–0.60	0.80–0.86	0.83–0.88
Reaching $\geq 80\%$	0.29	0.73	0.78
Lowest quarter (mean)	0.28	0.63	0.68
Highest quarter (mean)	0.83	0.96	0.97
Top–bottom gap	0.55	0.33	0.29
Cycles per student	0	2.41	2.29

Three patterns deserve comment. First, the loop did most of the work, and level matching added a smaller increment on top. Second, the loop narrowed the gap between the top and bottom quarters of students from 0.55 to 0.33 in the loop condition and 0.29 in the full condition. Students who start with unmastered objectives receive extra cycles, while strong students need few. The lowest quarter still lagged, so the loop narrows the gap without closing it.

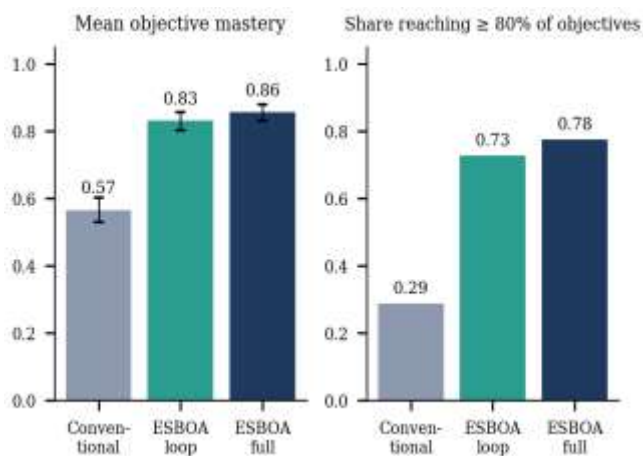


Figure 5. Mean mastery and share of students reaching the 80% threshold by condition. Error bars show the 95% interval across replications

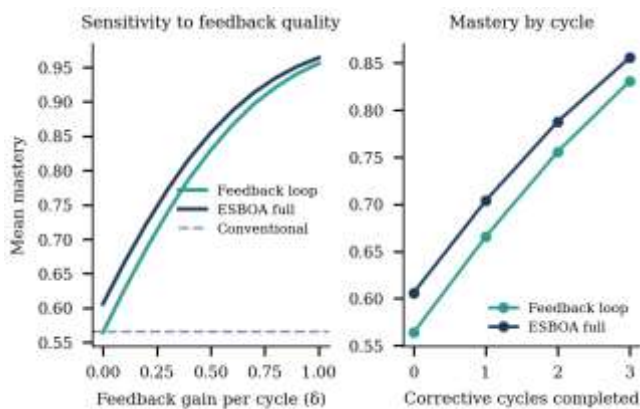


Figure 6. Sensitivity of mean mastery to feedback gain (left) and mastery after each corrective cycle (right)

Third, the benefit depends on feedback quality (Figure 6). When $\delta = 0$, meaning feedback carried no useful information, the loop gave no gain over the conventional course (0.57 against 0.57), yet students still needed 2.7 corrective cycles on average, so the effort was wasted. As δ rose, mastery rose and fewer cycles were needed. This is not an independent finding. In Equation 1 a cycle adds nothing when $\delta = 0$ by construction, so the zero-gain result mainly confirms that the code implements the model correctly. It is consistent with the wide variation in real feedback effects (Wisniewski et al., 2020), but it does not confirm it.

Sensitivity, Workload, and Interpretation

Table 7 repeats the comparison under alternative assumptions. The ordering conventional $\leq$ loop $\leq$ full held in every scenario, but the size of the gap changed a great deal, and in one scenario it vanished. Three results stand out. First, the gain from the loop depended mainly on feedback quality and on the cycle cap: it fell from 0.27 to 0.15 when δ was 0.25 and to zero when $\delta = 0$, and it rose from 0.19 to 0.36 as R went from 2 to 5, at a cost of 1.71 to 3.41 cycles per student. Second, the gain was fairly stable under changes to the ability distribution and difficulty range (0.24 to 0.30) and under heterogeneous feedback (0.25), so it is not an artefact of one ability distribution or one difficulty range. Third, the more realistic complications lowered absolute mastery without removing the gain: 10% assessment error reduced mean mastery in the loop condition from 0.83 to 0.78, 20% forgetting reduced it to 0.66 (and to 0.45 in the conventional course), and prerequisite dependency left a gain of 0.28. Raising the threshold to $\tau = 0.90$ cut the share of students reaching it to 0.10, 0.43, and 0.49 in the three conditions. The ordering of conditions is built into the model, so its stability is expected. What Table 7 adds is the size of the effect under each assumption and the strength of its dependence on feedback quality and the cycle cap.

Table 7. Mean Mastery Under Alternative Assumptions (One Change at a Time; 500 Replications of 200 Synthetic Students Each)

Scenario	Conv.	Loop	Full	Cycles (loop)
Base case (Table 1)	0.57	0.83	0.86	2.41
Feedback gain $\delta = 0.25$	0.57	0.71	0.75	2.57
Feedback gain $\delta = 0$	0.57	0.57	0.61	2.69
Heterogeneous feedback	0.57	0.82	0.84	2.39
Baseline gain $g = 0$	0.5	0.79	0.82	2.57
Discrimination $a = 0.8$	0.55	0.77	0.8	2.63
Mean ability -0.5	0.46	0.76	0.79	2.66
Ability SD 1.5	0.56	0.8	0.82	2.26
Difficulty -1.5 to 1.5	0.56	0.81	0.84	2.54
Cycle cap $R = 2$	0.57	0.76	0.79	1.71
Cycle cap $R = 5$	0.57	0.93	0.94	3.41
Assessment error 10%	0.57	0.78	0.81	2.34
Forgetting 20%	0.45	0.66	0.68	2.41
Prerequisite dependency	0.53	0.81	0.84	2.41

The workload implications are substantial. A student in the loop condition needed about 2.41 corrective cycles on average (2.29 in the full condition), equivalent to roughly six objective-level re-assessments per student, so a class of 200 would generate about 480 student-level cycles and about 1,200 re-assessments. Even after three cycles, 57% of simulated students (51% in the full condition) still had at least one unmastered objective and would be referred to the teacher. The cap therefore shifts a decision and time burden to teachers rather than removing it. Teacher workload, assessment burden, and instructional time would depend on how cheaply corrective tasks and re-assessments can be produced, for example through item banks, templates, and automated marking of routine items, and institutions would need to supply the extra assessment capacity. The cap trades mastery against workload: $R = 2$ reduced cycles to 1.71 per student but also reduced mean mastery to 0.76, whereas $R = 5$ raised mastery to 0.93 at 3.41 cycles. The pilot must measure these costs directly, and the cap and threshold are decisions for each department.

Taken together, the simulation supports three limited conclusions. The decision logic terminates and routes cases as designed; the benefit of the loop is conditional on informative feedback and sufficient cycles; and the loop carries a workload cost. Because ability, difficulty, and gains are fixed by assumption, the numbers should not be quoted as expected effects. Several features of real learning remain missing or crudely represented: forgetting is a single retention check, dependencies between objectives are a simple penalty, assessment error is symmetric, and variation in how teachers implement the process is not modelled.

These limits constrain how far the results can be read, and they are repeated in Limitations and implications.

Pilot Roadmap, Indicators, and Risks

This section describes a plan. Nothing in it has been implemented or tested, and none of it is a result. The proposed pilot starts in selected courses of one university computer science department, for example Programming I, Java Programming, Data Structures and Algorithms, Database Systems, and Research Methodology. The final choice will be made with the university and the instructors. A twelve-month plan (Figure 7) moves from literature review and framework refinement through instrument development, course selection, and teacher preparation to a three-month implementation with data collection, followed by analysis, framework improvement, final evaluation, and publication.

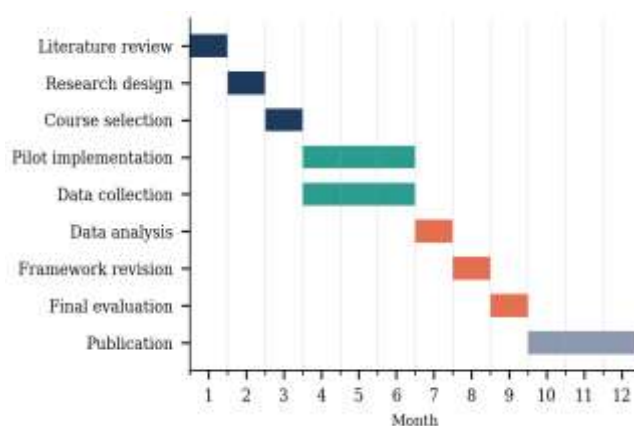
**Figure 7.** Proposed twelve-month pilot timeline

Table 8. Planned Evaluation Indicators

Strand	Indicator	Planned source
Quantitative	Pre-test and post-test performance	Course tests
Quantitative	Share of learning objectives achieved	Objective-level scores
Quantitative	Alignment of assessments with objectives	Assessment map
Quantitative	Assignment completion and project performance	Course records
Quantitative	Progression, participation, time on selected activities	Course records
Qualitative	Student and teacher satisfaction	Questionnaires
Qualitative	Quality of feedback and implementation problems	Interviews, observation
Management	Completeness of educational records; feasibility of scaling	Records, management interviews

Table 9. Risk Register

Risk	Planned response
Resistance to a new process	Training and stakeholder participation
Extra teacher workload at the start	Templates, planning support, gradual rollout
Inconsistent implementation	Clear procedures and monitoring
Insufficient data	Standardised data collection
Technical limitations	Begin with a low-cost prototype
Limited pilot duration	Realistic evaluation indicators
Difficulty comparing results	Baseline measures before implementation

Table 8 lists the indicators to be fixed before the pilot begins. They are research measures, not assumed outcomes. Implementation will meet obstacles. Table 9 sets out the risks identified so far and the planned responses.

Ethical safeguards are part of the design and not an add-on. The pilot needs institutional approval before data collection, informed participation where required, confidentiality and secure handling of student data, transparent evaluation procedures, avoidance of unfair automated decisions, and human oversight of educational decisions. ESBOA is meant to support teachers and managers, and its outputs should be treated as evidence for a decision, not as the decision itself.

Limitations and Implications

This paper has clear limits. It presents a design and a simulation, not classroom evidence. The simulation parameters are assumptions, and real learning is messier than a logistic curve: students forget, objectives depend on one another, and feedback quality differs from teacher to teacher. The sensitivity analyses in Table 7 probe these features crudely but do not replace empirical calibration, and the model still omits variation in how teachers implement the process. The framework itself has not been reviewed by external experts or evaluated against structured design criteria, and its problem list was not drawn from a systematic requirements study. Its novelty is also relative, because most of its components are established. Mastery-style loops have a mixed evidence base (Kulik et al., 1990; Slavin, 1987), so the pilot should check effects on durable learning and not only on repeated tests of the same objectives. A highly structured process may

reduce unwanted variation, but too much standardization could limit teacher creativity. Finally, the first scope is computer science education, where algorithmic thinking is familiar, and transfer to other disciplines should wait for evidence.

For teachers, ESBOA offers a checklist-like way to keep objectives, assessment, and feedback consistent. For departments, it offers a record structure for monitoring. For researchers, it offers hypotheses that can be tested: that explicit links between objectives and assessment, bounded corrective loops, and structured records improve the clarity, consistency, and measurability of teaching. Future work should obtain expert review of the framework, compare it with existing mastery-learning and outcome-based designs, refine thresholds and cycle caps, study teacher workload, and examine whether generative AI tools can help produce specific feedback under teacher oversight (Kasneci et al., 2023).

Conclusion

ESBOA treats education as a structured, measurable, and feedback-driven process: Input, Objective, Algorithm/Instructions, Teaching, Learning, Assessment, Feedback, and Output, with planning, monitoring, reporting, and improvement across the lifecycle. It remains a proposed framework that has been examined computationally; its educational effectiveness has not been established. The contribution of this paper is conceptual and computational: an explicit architecture and set of principles, a bounded corrective-learning algorithm with teacher referral, and a simulation protocol that separates verification from validation. In simulation, the loop improved mastery

and narrowed gaps only under assumed gains, only when feedback was informative, and at a workload cost that grew with the cycle cap. These findings describe the behaviour of a model, not of real students, and they do not show improved teaching quality, learning outcomes, or educational management. Empirical validation is the necessary next step: expert evaluation of the framework, classroom implementation in computer science courses, comparative studies against existing mastery-learning and outcome-based designs, and assessment of learning outcomes, teacher workload, and institutional feasibility, with teachers remaining in charge of decisions.

Acknowledgments

The author thanks Bakhtar University for their intellectual support.

Author Contributions

Conceptualization, methodology, software, formal analysis, visualization, writing—original draft preparation, and writing—review and editing, S.M.K.R and MH. The author has read and agreed to the published version of the manuscript.

Funding

This research received no external funding.

Conflicts of Interest

The author declares no conflict of interest.

References

- Azevedo, B. F., Amoura, Y., Kantayeva, G., Pacheco, M. F., Pereira, A. I., & Fernandes, F. P. (2021, July). Collaborative Learning Platform Using Learning Optimized Algorithms. In *International Conference on Optimization, Learning Algorithms and Applications*. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-91885-9_52
- Biggs, J., & Tang, C. (2011). *Teaching for quality learning at university: What the student does* (4th ed.). McGraw-Hill/Open University Press.
- Biggs, J. (1996). Enhancing teaching through constructive alignment. *Higher Education*, 32(3), 347–364. <https://doi.org/10.1007/BF00138871>
- Black, P., & Wiliam, D. (1998). Assessment and classroom learning. *Assessment in Education: Principles, Policy & Practice*, 5(1), 7–74. <https://doi.org/10.1080/0969595980050102>
- Bond, M., Khosravi, H., De Laat, M., Bergdahl, N., Negrea, V., Oxley, E., Pham, P., Chong, S. W., & Siemens, G. (2024). A meta systematic review of artificial intelligence in higher education: A call for increased ethics, collaboration, and rigour. *International Journal of Educational Technology in Higher Education*, 21. <https://doi.org/10.1186/s41239-023-00436-z>
- Gregor, S., & Hevner, A. R. (2013). Positioning and presenting design science research for maximum impact. *MIS Quarterly*, 37(2), 337–355. <https://doi.org/10.25300/MISQ/2013/37.2.01>
- Guskey, T. R. (2007). Closing achievement gaps: Revisiting Benjamin S. Bloom's "Learning for Mastery". *Journal of Advanced Academics*, 19(1), 8–31. <https://doi.org/10.4219/jaa-2007-704>
- Harden, R. M. (2007). Outcome-based education: The future is today. *Medical Teacher*, 29(7), 625–629. <https://doi.org/10.1080/01421590701729930>
- Hattie, J., & Timperley, H. (2007). The power of feedback. *Review of Educational Research*, 77(1), 81–112. <https://doi.org/10.3102/003465430298487>
- Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, 28(1), 75–105. <https://doi.org/10.2307/25148625>
- Holmes, W., Porayska-Pomsta, K., Holstein, K., Sutherland, E., Baker, T., Shum, S. B., ... Koedinger, K. R. (2022). Ethics of AI in education: Towards a community-wide framework. *International Journal of Artificial Intelligence in Education*, 32(3), 504–526. <https://doi.org/10.1007/s40593-021-00239-1>
- Ifenthaler, D., & Yau, J. Y.-K. (2020). Utilising learning analytics to support study success in higher education: A systematic review. *Educational Technology Research and Development*, 68(4), 1961–1990. <https://doi.org/10.1007/s11423-020-09788-z>
- Kabudi, T., Pappas, I., & Olsen, D. H. (2021). AI-enabled adaptive learning systems: A systematic mapping of the literature. *Computers and Education: Artificial Intelligence*, 2, 100017. <https://doi.org/10.1016/j.caeai.2021.100017>
- Kasneci, E., Sessler, K., Küchemann, S., Bannert, M., Dementieva, D., Fischer, F., ... Kasneci, G. (2023). ChatGPT for good? On opportunities and challenges of large language models for education. *Learning and Individual Differences*, 103, 102274. <https://doi.org/10.1016/j.lindif.2023.102274>
- Kulik, C.-L. C., Kulik, J. A., & Bangert-Drowns, R. L. (1990). Effectiveness of mastery learning programs: A meta-analysis. *Review of Educational Research*, 60(2), 265–299. <https://doi.org/10.3102/00346543060002265>
- Kulik, J. A., & Fletcher, J. D. (2016). Effectiveness of intelligent tutoring systems: A meta-analytic review. *Review of Educational Research*, 86(1), 42–78. <https://doi.org/10.3102/0034654315581420>

- Nicol, D. J., & Macfarlane-Dick, D. (2006). Formative assessment and self-regulated learning: A model and seven principles of good feedback practice. *Studies in Higher Education*, 31(2), 199–218. <https://doi.org/10.1080/03075070600572090>
- Peffer, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems*, 24(3), 45–77. <https://doi.org/10.2753/MIS0742-1222240302>
- Rutten, N., van Joolingen, W. R., & van der Veen, J. T. (2012). The learning effects of computer simulations in science education. *Computers & Education*, 58(1), 136–153. <https://doi.org/10.1016/j.compedu.2011.07.017>
- Sadler, D. R. (1989). Formative assessment and the design of instructional systems. *Instructional Science*, 18(2), 119–144. <https://doi.org/10.1007/BF00117714>
- Sargent, R. G. (2013). Verification and validation of simulation models. *Journal of Simulation*, 7(1), 12–24. <https://doi.org/10.1057/jos.2012.20>
- Shute, V. J. (2008). Focus on formative feedback. *Review of Educational Research*, 78(1), 153–189. <https://doi.org/10.3102/0034654307313795>
- Slade, S., & Prinsloo, P. (2013). Learning analytics: Ethical issues and dilemmas. *American Behavioral Scientist*, 57(10), 1510–1529. <https://doi.org/10.1177/0002764213479366>
- Slavin, R. E. (1987). Mastery learning reconsidered. *Review of Educational Research*, 57(2), 175–213. <https://doi.org/10.3102/00346543057002175>
- Venable, J., Pries-Heje, J., & Baskerville, R. (2016). FEDS: A framework for evaluation in design science research. *European Journal of Information Systems*, 25(1), 77–89. <https://doi.org/10.1057/ejis.2014.36>
- Wang, X., Su, Y., Cheung, S., Wong, E., & Kwong, T. (2013). An exploration of Biggs' constructive alignment in course design and its impact on students' learning approaches. *Assessment & Evaluation in Higher Education*, 38(4), 477–491. <https://doi.org/10.1080/02602938.2012.658018>
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35. <https://doi.org/10.1145/1118178.1118215>
- Winget, M., & Persky, A. M. (2022). A practical review of mastery learning. *American Journal of Pharmaceutical Education*, 86(10), 8906. <https://doi.org/10.5688/ajpe8906>
- Wisniewski, B., Zierer, K., & Hattie, J. (2020). The power of feedback revisited: A meta-analysis of educational feedback research. *Frontiers in Psychology*, 10, 3087. <https://doi.org/10.3389/fpsyg.2019.03087>
- Zawacki-Richter, O., Marín, V. I., Bond, M., & Gouverneur, F. (2019). Systematic review of research on artificial intelligence applications in higher education: Where are the educators? *International Journal of Educational Technology in Higher Education*, 16, 39. <https://doi.org/10.1186/s41239-019-0171-0>