Towards Didactics-Driven Development of Educational Games

Anders Bouwer
Smart Education Lab
Amsterdam University of Applied Sciences
Amsterdam, Netherlands
a.j.bouwer@hva.nl

Riemer van Rozen
Centrum Wiskunde & Informatica
Amsterdam, Netherlands
rozen@cwi.nl

Abstract

Educational games offer an effective means to enhance interactive learning experiences. However, developing high quality educational games is difficult. In particular, integrating didactic goals into a game's design, and verifying the learning outcomes is a complex iterative process. Due to a lack of control over these concerns, fully developed games may lack the intended educational value.

We aim to improve educational game development by structuring and partly automating this process. We propose Didactics-Driven Development, a novel framework for keeping didactic goals and concerns in focus throughout the development process. By making didactic concerns explicit, it enables testing interaction patterns against opportunities for learning. We discuss how the approach has been applied to three case studies and how this has informed ongoing development of the framework.

CCS Concepts

• **Software and its engineering** → **Domain specific languages**; **Integrated and visual development environments**; • **Applied computing** → **Computer games**; **Education**; **Interactive learning environments**.

Keywords

Didactics-driven development, educational game development, game design framework, game tutorials, learning opportunities, playtraces

ACM Reference Format:
Anders Bouwer and Riemer van Rozen. 2025. Towards Didactics-Driven Development of Educational Games. In *International Conference on the Foundations of Digital Games (FDG '25), April 15–18, 2025, Graz, Austria.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3723498.3723847

1 Introduction

This paper proposes a framework called Didactics-Driven Development, for incorporating didactic concerns into the design and development of educational games. It addresses the problem that in current practice, didactic goals are mostly considered at the start and the end of the process of developing an educational game, but often do not receive enough attention during the iterative design cycles in between. This lack of attention can cause a mismatch between the intended learning outcomes and actual learning experience, leading to delays in the development process, or even worse, an educational game of suboptimal quality. The goal of the Didactics-Driven Development framework is to prevent this mismatch from occurring, by making explicit the didactic concerns involved and their relationships with design decisions during the whole development process of an educational game. This helps to ensure that (1) opportunities for learning related to the didactic goals are in fact included in the educational game design; (2) the system can check whether these opportunities will also occur during actual gameplay; and (3) the system can measure to what extent a player takes advantage of these learning opportunities during the gameplay experience, by recognizing these opportunities and relating them to interaction patterns in the playtrace representing the player's behaviour during the gameplay.

The framework has developed over the course of three case studies described in this paper. The framework started out as a conceptual design aid, and has been used as such to inform the design of two educational games. Key parts of the framework have been implemented in a working game tutorial application to demonstrate the technical feasibility of the approach.

The rest of this paper is structured as follows: Section 2 describes related work in the fields of educational game design, procedural content generation, and software development; Section 3 lists the research questions; Section 4 describes the methods used, and how the case studies have been instrumental in developing the didactics-driven development framework; Section 5 describes the framework in detail; Section 6 presents three case studies in which the framework has been used; Section 7 discusses current limitations and suggestions for future work; Section 8 concludes this paper.

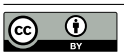
2 Related Work

This section describes related work from three perspectives: educational game design, procedural content generation, and software development.

2.1 Educational game design

Developing educational games is hard. There is no proven method to successfully design good games, let alone games which can address specific learning objectives. To support educational game designers, several frameworks, tools and models have been proposed in the literature. Without trying to be complete, the following overview lists a few of the approaches.

Design models are tools to support the ideation phase, e.g., card decks like Archambault's Game of Games [6] or Schell's design card deck of lenses [21]. These offer suggestions for elements to consider for inclusion or modification in the game design, thereby making the designer aware of the possible design space. However,



FDG '25, Graz, Austria

ACM ISBN 979-8-4007-1856-4/25/04
https://doi.org/10.1145/3723498.3723847

these tools offer no support for most of the work, from selecting the desired elements to applying and testing whether they work to achieve the goal.

The framework for the design and analysis of educational games by Aleven et al. [2] includes attention to three main components, of which the second is a framework in itself: (1) learning objectives [4]; (2) the MDA framework by Hunicke et al. [16]; and (3) instructional principles. This work recognizes the importance of didactic concerns, and shows how the elements of the framework have been applied in the (re)design of an educational game. However, it only offers support at the conceptual level, and little or no support for the connection between the three components.

Game ontologies support categorizing, understanding, and critical analysis in the field of game studies and game literacy (e.g., [13, 31]). This kind of work offers vocabulary for reasoning about this matter, some of which can be used to formalize, supporting the transition from the conceptual design level to software coding level. However, these ontologies are focused mostly on use in critical analysis and offer little support for the design process.

Design patterns for gameplay design [5, 10] or game mechanics [1, 12] connect higher level design goals to design patterns that can be translated into actual source code. This helps guarantee that the design goal can be achieved when the pattern can be applied. However, the proposed patterns so far are very specific, and do not comprise a general method for educational game design.

There are also practical methodologies for the design of educational serious games, which pay explicit attention to the design process. For instance, Weitze's Smiley model [29] explicitly combines terms from the field of education, such as "prerequisites for learning, learning goals, content, and evaluation/assessment", and terms from the field of game design, such as "game goal, action space, rules, choice, challenge and feedback" into one model, although not much support is given to how this mapping can be made. Silva's methodology [22] contains a flow chart indicating the main steps in defining a serious game (and their order, with some iterative loops towards the end), including selecting the topic and learning objectives; defining the target audience, learning style and environment; selecting a game genre, creating a story, scenarios, characters, and other elements; selecting [game] mechanics that can be used as learning mechanism, and other mechanics; defining the dynamics; resulting in game experiences such as learning outcomes and/or fun. However, the model does not offer support for the transition into game software development.

Authoring tools for the development or prototyping of educational games are often tailored towards a particular type of learning and gameplay, associated with specific genres of mini-games. Such tools offer a fast way to create small educational games, even without prior experience in game design or development. However, they are limited to creating basic games, and specific kinds of learning goals.

Technical frameworks for specific game genres are part of certain game engines. These can be used and modified by experts to offer more flexibility to develop other and more complex types of games. However, they require expertise in software development, both general, and specific for the game engine. Except for some work on reusable AI components for serious games [30], most of the authoring tools and technical frameworks do not offer explicit representations of game situations in terms of didactic concerns.

2.2 Procedural Content Generation

Within the field of Procedural Content Generation, several lines of work are relevant to our purpose. The work by Bogost on persuasion in games [11], and by Treanor et al. on procedural rhetorics [24] focuses on the aspects of communication in games. It does not give specific attention to didactic concerns, however.

Live Game Design aims to create live interactive visual models of a game that are directly connected to the game's code, that are easy to modify, so that design iterations of a prototype can be done much faster, ideally during runtime of the game [26, 27]. Although the approach introduces explicit relationships between design changes and their effects on gameplay for learning how to code, it does not model educational goals.

Procedural Content Generation (PCG) often serves to automate tasks of human game designers, but mixed-initiative approaches try to let both human and machine do things they are good at, in a more collaborative manner [17, 19, 23]. For instance, human designers can provide design goals, requirements, conditions, and initial sketches or ideas. The system can work out sketches, fill in details, give feedback to critique ideas, search for solutions that meet the requirements and constraints, or generate alternatives that explore neglected areas of the design space. The designer can reflect on, possibly modify, and select between suggested design alternatives, and control when and how computer support is desired.

Much work on PCG is controlled at the content level, which can be useful for well-specified and constrained tasks. However, when more flexibility is required, especially for educational contexts, it can be useful to manage what is being generated on a conceptually higher level, i.e. the level of design goals. Examples of this kind of work are sparse [12, 25]. Although these approaches are in principle generic and resuable, the actual implementations were done for specific games, and except for some work on game tutorials [7, 28], they do not address educational games or how didactic concerns can be realized.

2.3 Software development

In the field of software development, several methods relate to and have partly inspired the idea of didactics-driven development.

Requirements engineering methods (e.g., [3]) deal with the explicit specification of requirements for software. Bridging this kind of work to the realm of educational game development seems promising, since there are often many types of requirements involved. Agile software and game development methods rely on iterative cycles involving reflection and improvement rather than trying to get the product perfect in one go. However, there is little work in this area that focuses on educational game development, with some exceptions [18].

Test-driven development [8] is a well-established methodology in software engineering that starts with making explicit the goal of the software, so that the code to be written can be tested to see if it fulfills the goal. When combined with tools, this allows for automated generation and testing. However, it is usable only to the extent that the design goals were addressed in the requirements

analysis and can be expressed in a testable format [20]. For applied games designed with educational goals in mind, these goals are often lost in the translation from requirements into design and implementation choices, only to be addressed again towards the end of the development process.

To summarize, many frameworks for designing games exist, but some are focused only on the conceptual ideation phase, without support for more concrete game design and development. On the other hand, some frameworks and tools are focused more on the technical side of the software development, but lack the connection to the conceptual design goals. Some frameworks are very open-ended, and are more suited for analysis of existing games than for guiding the design process of new games.

We argue that what is necessary is a bridge between the design goals in a designer (team)'s mind, and their realization in a running prototype that can be easily tested and adapted to support a highly transparent and iterative method of improving an educational game in all phases of design, development, and maintenance.

3 Research questions

This study aims to answer the following research questions:

- How can designers of educational games be supported in the iterative improvement of the fit between their design goals and the player's gameplay and learning experience?
- How can didactic concerns be explicitly represented and related to game design decisions, learning opportunities, play traces, and insights for improving the design?
- How can didactic concerns be explicitly represented and related, so that educational game designers can define, track, and relate game design decisions, learning opportunities, play traces, and insights for improving the design?

4 Methods

The Didactics-Driven Development framework is a work in progress, following the method of Design Research [15], in which the framework has been iteratively created, tested, and refined in various case studies. Besides the related work mentioned in the previous section, it has also been informed by discussions with two professional game designers and three game development students who were all developing educational games in the context of the DGA FieldLab project Didactics-Driven Development[1]. Case study 1 was instrumental in forming many ideas in the framework. These were used in testing and refining the game's didactic goals and the relevant game content. In Case study 2 (as well as another student game prototype not discussed here), the framework played an important role as a conceptual aid in the design of the educational games from the start. In Case study 3, a subset of the most important activities in the framework have been implemented in the form of a tool for a game tutorial to see if the framework is specific enough to be programmed and whether such a tool can be useful for designers while playtesting and improving the game.

5 The Didactics-Driven Development Framework

The Didactics-Driven Development framework can serve different purposes. It can be used as a conceptual tool to aid in the design process, and parts of it have been implemented in the form of an executable design tool to keep track of didactic concerns during design and playtesting. As shown in Fig. 1, the Didactics-Driven Development Framework can serve as a conceptual aid, by providing an overview of designer actions during the design process, player actions in playtests, system support for the designer, and the different kinds of data involved. We see these concepts as content to be explicitly represented (including the relationships between them), so that it can be generated, traced, and analyzed.

Starting at the top-left of the figure, didactic goals and instructional principles inform design decisions and their rationale. The design process produces a model of the game's design that can be automatically transformed into code for a running game prototype. The game design model can also be compared to existing designs if they are available, e.g., from earlier similar projects; relevant existing designs can then be used as additional input for the design. As soon as possible, the game prototype can be playtested using a specific game scenario.

A game scenario typically addresses one or more selected didactic goal(s) and is represented as a partial specification of one or more game states (and a partial ordering of these if necessary) that relate to this/these goal(s). A game scenario can be handcrafted and matched to the selected didactic goals, or generated automatically, if algorithms for this task are available. For example, in the most elementary case, a game scenario could be generated by selecting or creating a game state in which the selected didactic goal is addressed by an exercise that requires the player to play a (sequence of) move(s) to reach the desired goal state. A game scenario can also include more complex evaluation measures, designed to gauge intended learning outcomes. For instance, if the didactic goals include practicing the skills involved in handling customer requests, a game scenario could be created that involves game states with actual customers and their requests, and evaluation measures that track a player's performance with respect to the intended outcomes (e.g., responding to at least 80% of customers, and handling 50% of their requests adequately in the game). The game scenario is a partial specification in the sense that it does not have to specify all details (e.g., about which customers or which requests) required in the actual gameplay, but it does specify the important aspects and constraints on the order in which they have to occur in the gameplay.

When the game is ready for testing with a game scenario, playtesting will result in play traces (records of game states, player actions, and system actions in the game), and feedback from players. This is already often done in common playtesting practice, but the framework adds an important element by having support from the system in generating *opportunities for learning* based on the game scenario. These learning opportunities can be realized, when a player applies the relevant knowledge and/or skills, indicating some level of mastery, or missed, when a player fails to act appropriately, giving no signs, or negative signs of mastering the desired knowledge or skill.

[1] https://didacticsdrivendevelopment.wordpress.com

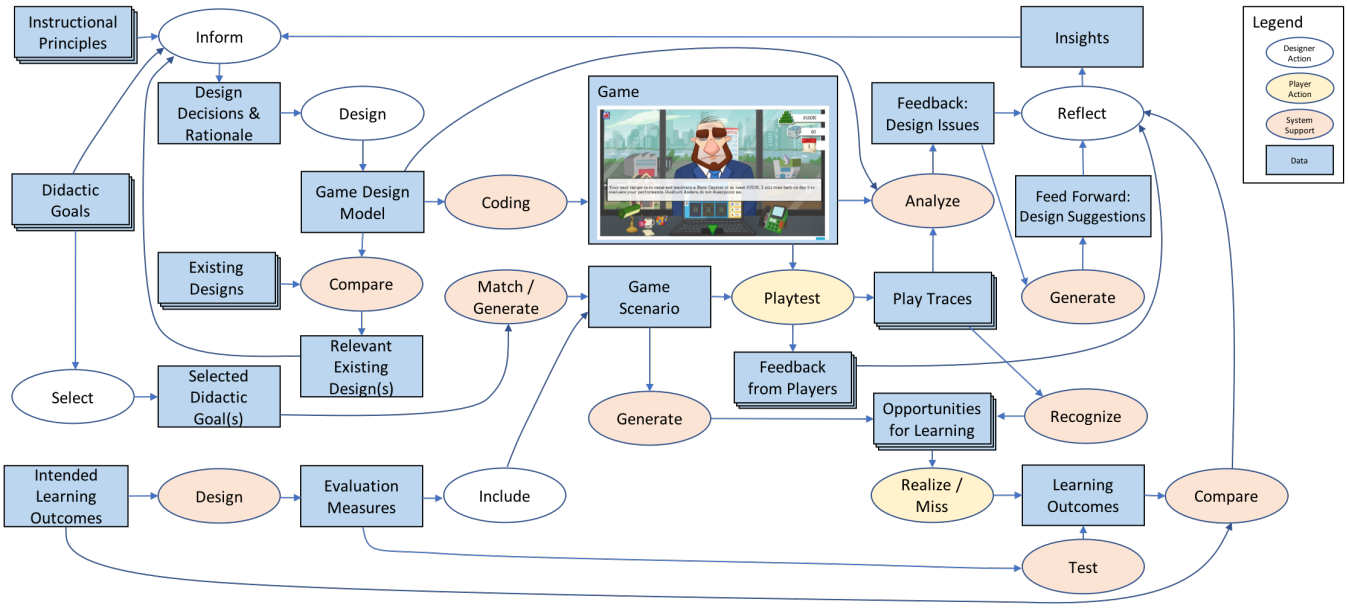


Figure 1: The Didactics-Driven Development Framework as a conceptual tool, applied to Case Study 1, a single-player game about how banks work. The legend indicates which parts are data, actions carried out by the designer or player, or actions that could be supported by an automated system.

Based on testing using information on the evaluation measures, the resulting learning outcomes can be compared to the intended learning outcomes.

The information from this comparison, along with feedback from players, serves as input for the designer to reflect upon, which may lead to further insights. In addition, the play traces may be analyzed to arrive at feedback on design issues, which may consequently be used as input to generate design suggestions as feed-forward. The resulting insights will feed back into informing new design decisions, along with a rationale for why they might improve the design.

The proposed Didactics-Driven Development framework has already been used to support the design process of several educational games. All of these games offer various pedagogical opportunities for experiencing and reflecting on the topics they address, at different times during and also after playing the game. To explain the potential of the framework, its application is described using three case studies involving one educational game released online (Case study 1), a multi-player educational game prototype (Case study 2), and one prototype game tutorial application (Case study 3).

6 Case Studies with the framework

6.1 Case Study 1. An educational game on the economy of banking

Case study 1 involved research on how the Didactics-Driven Development framework could benefit the development of Boom Bust Inc., an online single-player educational game on how banks work[2].

In the game, the user plays the role of a bank employee at the office, who has to decide to lend money, or not, to clients who ask for a loan. At various moments, the player receives instructions or feedback from the bank director and several other colleagues, who are Non-Player Characters in the game. The game included several didactic goals, including knowing what constitutes money in a bank account and where the money from bank loans comes from. During development, prototype versions of the game were playtested along with questionnaires asking about the play and learning experience, including questions about specific learning goals. This process led to various insights.

Insights from Case Study 1. Insights from applying the framework to this case study include the following:

- The game developers considered the method for distributing the game online with integrated pre- and post-test multiple-choice questionnaires as convenient and efficient.
- They expressed that it would be nice, however, if evaluation measures could be directly tied to gameplay instead of requiring an external multiple-choice questionnaire.
- Pre- and post-testing during development was very informative and showed both desired and undesired learning outcomes.
- For players, there were many learning goals, on different levels: the educational objectives, learning how to play the game, learning how to satisfy the boss, customers, and colleagues in the game, and persuasive goals, such as creating interest in alternative banking systems.
- These different goals and the dependencies between them were not all that clear before game development started, and should be made more explicit.

[2]https://pillargames.itch.io/boom-bust-inc

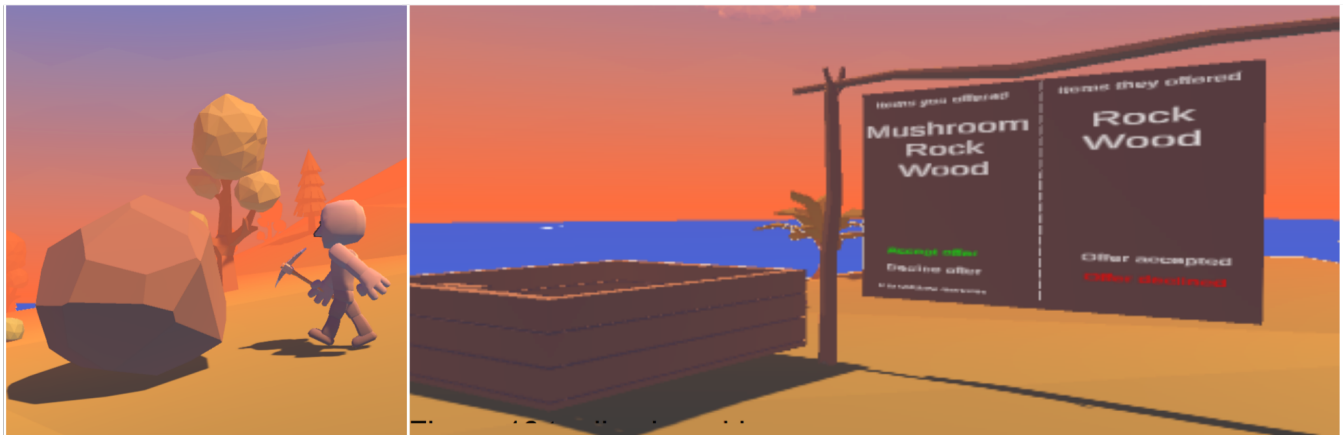


Figure 2: Multi-player island survival game in case study 2. On the left, a player is mining for resources and on the right, the interface for exchanging resources, such as food and building materials, is shown. Figure adapted from [9].

- There is a need to better document how these goals are related to game design decisions.
- The results were useful in improving the game design, in particular how to address the learning goal of what money in a bank account represents, and how players can gradually learn the different skills involved in the game.
- Discussions with game designers also informed the development of the framework. In particular, we realized the importance of explicitly representing the opportunities for learning that may take place during gameplay, and whether the learning outcomes match the intended ones.

6.2 Case Study 2. A multi-player game about collaboration, cooperation and competition

Case study 2 involves a multiplayer digital game (see Fig. 2) that allows collaboration, cooperation, and competition between players. The game's design allows scenarios such as starting alone on a part of an island where the player can start building small things with found resources, realizing that there are more people who have other kinds of resources that can be exchanged, and even another island that can be connected by building a bridge. It also includes a game master who can drop resources and/or weapons at specific locations, distribute information, or assign goals to players. The game is meant to serve as a educational and research test pad for exploring how game design decisions, or actions by the game master, affect individual and group gameplay and player experiences. When complemented with debriefing and discussion afterwards, it can be used to make players more aware of how their (perhaps competitive, or even violent) gaming behaviour was triggered and their gameplay experience affected by particular events in the game, other players' behaviour, and/or conscious decisions of game designers.

The three types of player behaviour are defined as follows:

- Collaboration: work together towards a common goal, e.g. build facilities that benefit all, or move a big rock together;
- Cooperation: work together with benefits for each player E.g., one player gathers rocks, while another gathers wood;
- Competition: players compete for benefits. Rivalry between players within a team, or between teams of different islands, e.g., by stealing resources.

The game master can fly around the two islands in the game world unnoticed to observe players, communicate with them, and influence aspects of the game world in order to stimulate collaboration, cooperation, or competition during gameplay.

Insights from Case Study 2. Insights from this case study include the following:

- A game which allows for different scenarios that address different didactic goals can benefit from or may even require a game master to decide on a particular scenario and how this addresses particular goals.
- In this game, didactic expertise is used broadly in the design of the game world, player and team goals, and required by the game master.
- In a relatively open multi-player game such as this one, planning for learning needs to be opportunistic. For instance, rather than aiming for one specific learning goal, which may be easily missed, it makes more sense to take advantage of learning opportunities when they occur.
- Didactic opportunities can occur:
  - during gameplay: when players choose to collaborate, cooperate or compete;
  - when players meet, find resources or problems;
  - when the game master sets up didactic opportunities by manipulating goals, problems and resources on the fly;
  - after the game: a discussion with the game master can help players reflect on the players and teams' collaborating, cooperating and competing behaviours and the reasons behind them.
- Categorizing game actions as collaborative, cooperative or competitive made it easy to recognize these types of behaviour in playtraces of recorded gameplay.
- Reflecting on missed opportunities for cooperation and collaboration was considered important for learning about the benefits of such behaviour.

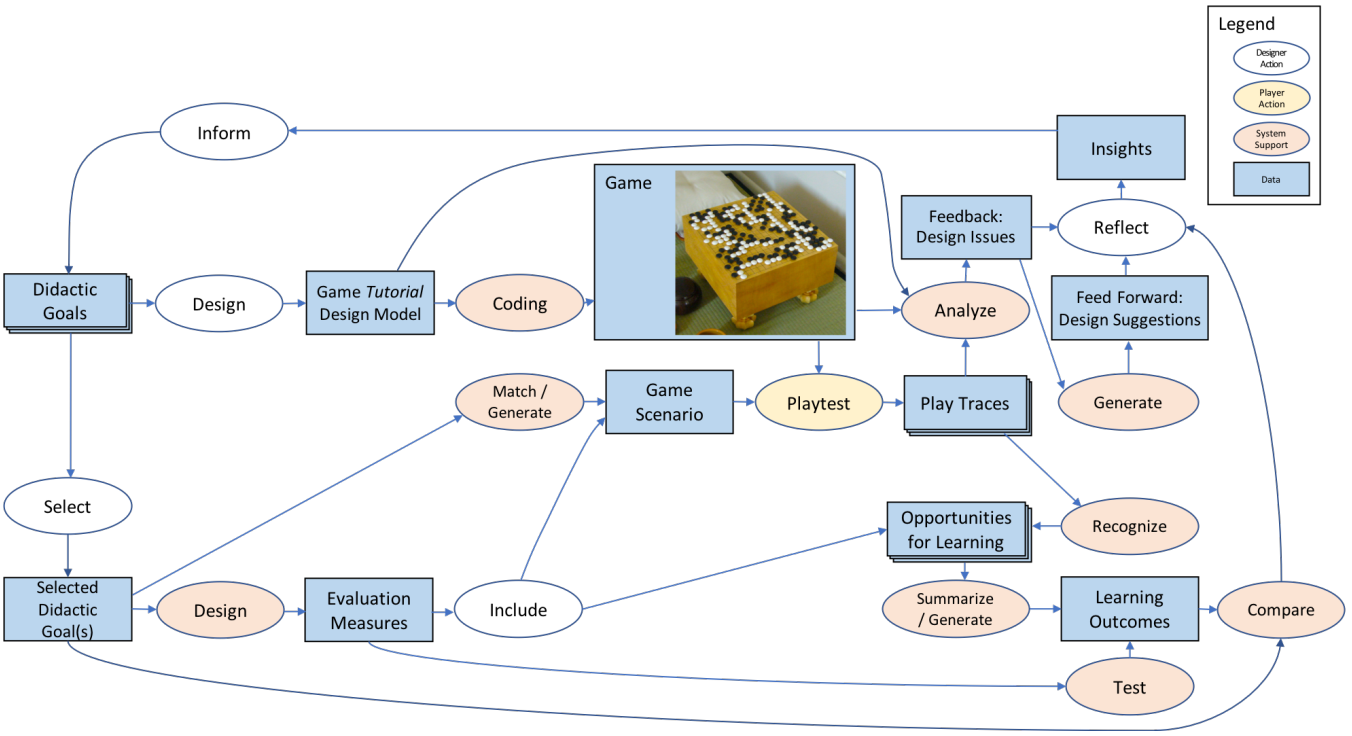


Figure 3: The parts of the Didactics-Driven Development Framework that have been applied and implemented in Case Study 3 for a Go game tutorial.

- Recognizing missed opportunities for cooperation or collaboration was done manually by the game master, but could be automated to some degree by considering the prequisite relationships between game actions.
- Recording game masters' actions and discussing their reasons for taking them could inform the design of automated support for game control in the future.

6.3 Case Study 3. A Tutorial for Learning how to play Go

In order to test whether the relationships between the concepts in the Didactics-Driven Development framework can also be used in an automated manner, a prototype interactive tool has been implemented that supports a subset of the tasks in the framework for a particular tutorial game, as shown in Fig. 3. The tutorial addresses learning goals for different types of skills in the classic board game Go, such as recognizing danger and opportunities to fight and stay alive. The tool makes explicit several types of didactic concerns involved in the design process, in particular didactic goals, intended learning outcomes, learning opportunities, play traces, and evidence for learning outcomes during gameplay, as shown in Fig. 4. The system is aware of the game scenario and didactic goal, recognizes learning opportunities, records the play trace, and tracks the evaluation measures. Currently, simple formulas are used to evaluate whether a didactic goal has been met, incorporating variables such as the number of relevant exercises the user has attempted and whether correct moves were played or not. Not shown in Fig. 4 are the final steps leading to insights and suggestions which can be fed back to the game designer. Insights can for example relate to missed learning opportunities, or exercises that are considered too difficult based on playtest data, and suggestions can for example be to consider revising an exercise, or the order of exercises.

Some parts of the framework were deliberately left out of this case study, such as how instructional principles informed the design, the comparison of the game design model to other existing designs, and dealing with player feedback. This was done to reduce complexity while retaining a subset of processes that reflects the cyclic nature of the framework.

Insights from Case Study 3. Insights from this case study include the following:

- Initial attempts at implementing the framework for the games of case study 1 and 2 failed because the task seemed too complex, and only isolated parts, such as learning goals or game scenarios could be represented explicitly in game code. Human designers may conceptually understand what is meant by a learning opportunity or design issue, but representing this more formally is still a struggle.
- To be able to implement the framework, we needed a game with simple rules and clear representation. To clarify how the framework works, we should explain how it applies to a well-known game. For these reasons, we excluded the games of case study 1 and 2, and focused on the game of Go instead.
- In a tutorial context about the elementary rules of Go, a game scenario can be defined as a game state (the go board with

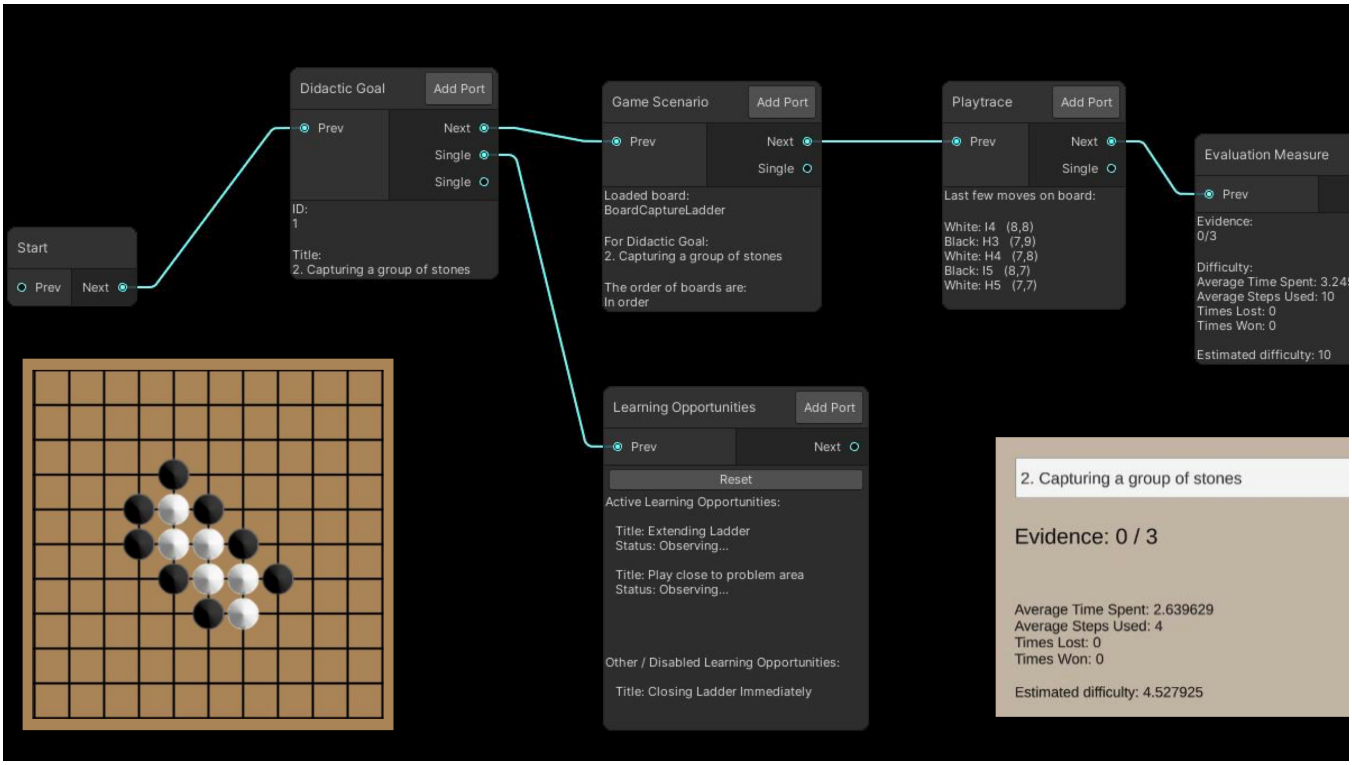


Figure 4: A situation in a tutorial game of Go that applies to learning about how to end ladders, and the underlying representations in the tool until this moment in the game. Figure adapted from [14].

moves played so far) in which particular rules are relevant in (one of) the next move(s).
- Didactic opportunities for learning the most important rules of the board game Go can be detected automatically and acted upon by the prototype during gameplay.
- The learner can learn the game's rules and how to apply them directly in the relevant game context while playing.
- The Didactics-Driven Development framework is not only useful as a conceptual tool, but can also be used as a basis for implementation of parts of the process.
- The prototype can be used to illustrate the workings of the Didactics-Driven Development framework as it shows the active part of the framework at each moment during the process of actual gameplay, complemented by the actions involving analysis and feedback generation of the prototype.

7 Discussion

Using the framework during the development of educational games in the three case studies described has led to a greater awareness of the didactic concerns during the design process, and resulted in games that are more flexible in that they can support multiple learning goals following different teaching scenarios. In case study 1, the framework stimulated efforts to define and differentiate the learning goals more clearly, to evaluate the learning goals, to recognize design flaws in addressing some of these goals, and discussion of alternative methods of evaluation, more directly tied to actual gameplay. In case study 2, the framework was used to inform the design of the game from the start, making it flexible enough to experiment with different game scenarios that might address different learning goals. These game scenarios can unfold involving competitive, cooperative, or collaborative player behaviour that can be detected in playtraces, or by observation by a game manager. In the other direction, work on case studies 1 and 2 has helped to formulate and refine the framework, respectively. In case study 3, the goal of implementing a substantial part of the framework was achieved. This allows players to learn about the basic rules of Go while playing, designers to gain feedback about actual vs. intended learning outcomes, and researchers to understand how the framework works.

Use of the framework by other game developers and researchers outside our network will be necessary to test its value for other users. As this is work in progress, we welcome feedback on the framework from other people working on the same problem. Working with and refining the framework further is expected to lead to more and better information about learning outcomes and other didactic concerns at different times during the development process of educational games.

The prototype tool is still under development, but can already be used to interactively illustrate several processes in the Didactics-Driven Development framework for a particular tutorial game.

7.1 Limitations

Although the framework is generic and can in principle be applied to any educational game that can be characterized in terms of didactic goals, game states, and learning opportunities, the current implementation only works for case study 3, i.e., the tutorial for Go. Implementations of the framework that could automate processes related to other case studies are expected to be possible but require manual implementation effort, especially when the game involves other kinds of game mechanics, or other pedagogical strategies. When the representation of learning goals, game states, or game scenarios becomes more complex, as in more open-ended game worlds, this is not trivial, however.

7.2 Future Work

Future work will include the following: (1) adapting the framework to other games, and refining it further, based on the results. We may start with other types of board games, or other restricted domains of games (e.g., PuzzleScript games) and try to generalize from there; (2) test and improve the usability of the framework and prototype tools with other game designers; (3) explore to what extent and how different instructional principles can be integrated into the implementation of the framework; (4) explore how the idea of comparing a game design model to existing designs can be integrated into the implementation of the framework, inspired by relevant work in this direction [25]; (5) further develop expertise on the intersection of educational game design, software engineering using domain-specific languages, and (game) didactics; and (6) create reusable tools for game designers that make it easier to integrate didactic concerns into the design process and resulting game code.

8 Conclusion

In order to better deal with the integration of didactic goals and other concerns in the development process of educational games, this paper has introduced a new framework, called Didactics-Driven Development. The framework builds on related work in the fields of educational game design, procedural content generation, game design patterns, and agile software engineering. Although still work in progress, use of the Didactics-Driven Development framework as a conceptual tool has been valuable during the development of a single-user game about the economics of banking, a multi-user game that can be used to learn about collaboration, cooperation, and competition, and a tutorial for the basic rules of Go.

Consequently, the resulting games are better suited to the didactic concerns involved, and more flexible, offering opportunities for learning for different teaching scenarios. The games can be used to address specific learning objectives, with relevant learning opportunities that are integrated into the gameplay.

The prototype developed for the tutorial about Go demonstrates that many of the concepts in the framework, such as didactic goals, game scenarios, opportunities for learning, learning outcomes, evaluation measures, play traces, and feedback on design issues, and the relationships between these concepts, can be modelled and reasoned about explicitly in a working software tool. The prototype can be used interactively by a player and game designer in the context of playtesting to improve the tutorial design. Further work will address limitations of the prototype by adapting and generalizing it to other games, and testing its usability and usefulness with other game designers. The framework will need to be evaluated and refined as well, in order to create reusable tools for game designers to integrate didactic concerns into the design process and game code.

Acknowledgments

We thank the Dutch Games Association and CLICKNL for funding part of this work as a DGA Field Lab project.

We would like to thank Paul Brinkkemper for discussions on Didactics-Driven Development and letting us use the game Boom-Bust Inc. by Pillar Games in case study 1, Marcus Bicker Caarten for his work on the prototype game for case study 2 [9], and Marc Gomersbach for his work on the prototype for case study 3 [14].

We also thank the anonymous reviewers for their insightful feedback that helped improve this paper.

References

[1] E. Adams and J. Dormans. 2012. *Game Mechanics: Advanced Game Design.* New Riders Publishing.

[2] V. Aleven, E. Myers, M. Easterday, and A. Ogan. 2010. Toward a framework for the analysis and design of educational games. In *Proceedings of third IEEE international conference on digital game and intelligent toy enhanced learning, 2010.* IEEE.

[3] R. Ali, F. Dalpiaz, and P. Giorgini. 2010. A goal-based framework for contextual requirements modeling and analysis. *Requirements Engineering* 15, 4 (2010), 439–458.

[4] L. W. Anderson and D. R. Krathwohl. 2001. *A taxonomy for learning, teaching, and assessing: A Revision of Bloom's Taxonomy of Educational Objectives.* New York: Longman.

[5] A. Antonaci, R. Klemke, and M. Specht. 2015. Towards design patterns for augmented reality serious games. In *Proceedings of mLearn 2015, CCIS, vol. 560.* Springer, Cham.

[6] C. Archambault. 2024. Game of Games: Game-Design Based Learning for All. In *Game Demonstrations, Presented at European Conference on Game-Based Learning, 2-4 October 2024, Aarhus, Denmark.*

[7] B. Aytemiz, I. Karth, J. Harder, A. M. Smith, and J. Whitehead. 2018. Talin: A Framework for Dynamic Tutorials Based on the Skill Atoms Theory. In *Proceedings of the 14th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, AIIDE 2018.* AAAI Press.

[8] K. Beck. 2002. *Test-Driven Development by Example.* Addison Wesley.

[9] M. Bicker Caarten. 2020. *Collaborative, cooperative and competitive player behaviour within a real time multiplayer survival game.* Bachelor's thesis. HBO-ICT, Game Development, Amsterdam University of Applied Sciences.

[10] S. Björk and J. Holopainen. 2005. Games and design patterns. *The game design reader: A rules of play anthology* (2005), 410–437.

[11] I. Bogost. 2010. *Persuasive games: The expressive power of videogames.* MIT Press.

[12] R. Corstjens, A. Bouwer, J. Dormans, and R. van Rozen. 2018. Wonderful Design: Applying Appraisal Theory to Procedural Level Generation. In *Proceedings of the 5th Workshop on Experimental AI in Games, EXAG 2018, at the AIIDE Conference, Edmonton, Canada, Nov. 13–17, 2018.*

[13] C. Elverdam and E. Aarseth. 2007. Game Classification and Game Design: Construction Through Critical Analysis. *Games Cult.* 2, 1 (2007).

[14] M. Gomersbach. 2023. *Learning Go through the DDD Methodology: An implementation of the Didactics-Driven Development Methodology to tutorialize Go.* Bachelor's thesis. HBO-ICT, Game Development, Amsterdam University of Applied Sciences.

[15] A. R. Hevner, S. Ram, T. M. Salvatore, and J. Park. 2004. Design Science in Information Systems Research. *MIS Quarterly* 28, 1 (March 2004).

[16] R. Hunicke, M. LeBlanc, and R. Zubek. 2004. MDA: A formal approach to game design and game research. In *Proceedings of the AAAI Workshop on Challenges in Game AI, 2004.* AAAI Press.

[17] D. Karavolos, A. Bouwer, and R. Bidarra. 2015. Mixed-Initiative Design of Game Levels: Integrating Mission and Space into Level Generation. In *Proceedings of the 10th International Conference on the Foundations of Digital Games 2015 (FDG 2015), June 22-25, 2015, Pacific Grove, CA.*

[18] L. J. Kortmann and C. Harteveld. 2009. Agile game development: Lessons learned from software engineering. In *Learn to game, game to learn, Proceedings of the 40th ISAGA Conference, 2009.*

[19] G. Lai, F. F. Leymarie, and W. Latham. 2022. On Mixed-Initiative Content Creation for Video Games. *IEEE Transactions on Games* 14, 4 (2022).
[20] L. Madeyski. 2010. *Test-Driven Development - An Empirical Evaluation of Agile Practice.* Springer.
[21] J. Schell. 2019. *The Art of Game Design: A Book of Lenses, 3rd Edition.* CRC Press.
[22] F. G. Silva. 2019. Practical Methodology for the Design of Educational Serious Games. *Information* 11, 1 (2019).
[23] G. Smith, J. Whitehead, and M. Mateas. 2010. Tanagra: A Mixed-Initiative Level Design Tool. In *Proceedings of the 5th International Conference on the Foundations of Digital Games, FDG 2010, Monterey, CA, USA, June 19–21, 2010.*
[24] M. Treanor and M. Mateas. 2013. An Account of Proceduralist Meaning. In *Proceedings of the 2013 DiGRA Int. Conference: DeFragging Game Studies, DiGRA 2013, Atlanta, GA, Aug. 26–29, 2013.* Digital Games Research Association.
[25] R. van Rozen. 2015. A Pattern-Based Game Mechanics Design Assistant. In *Proceedings of the 10th International Conference on the Foundations of Digital Games, FDG 2015.* SASDG.
[26] R. van Rozen. 2025. Live Game Design: Prototyping at the Speed of Play. In *Proceedings of the 20th International Conference on the Foundations of Digital Games, FDG 2025.* ACM.
[27] R. van Rozen and T. van der Storm. 2019. Toward live domain-specific languages: From text differencing to adapting models at run time. *Software & Systems Modeling* 18 (2019).
[28] D. Vet and R. van Rozen. 2024. The Puzzle Forecast: Tutorial Analytics Predict Trial and Error. In *Proceedings of the 19th International Conference on the Foundations of Digital Games, FDG 2024.* ACM.
[29] C. L. Weitze. 2016. Designing for Learning and Play - The Smiley Model as a Framework. *Interaction Design and Architecture(s)* 29, 1 (2016).
[30] W. Westera, R. Prada, S. Mascarenhas, P. A. Santos, J. Dias, M. Guimarães, ..., and C. Christyowidiasmoro. 2019. Artificial intelligence moving serious gaming: Presenting reusable game AI components. *Education and Information Technologies* (2019), 1–30.
[31] J. P. Zagal, M. Mateas, C. Fernández-vara, B. Hochhalter, and N. Lichti. 2007. Towards an Ontological Language for Game Analysis. In *Worlds in Play, International Perspectives on Digital Games Research, Ed. by S. de Castell and J. Jenson. Peter Lang, 2007.*