IJC International Journal of Computer
Peer-Reviewed International Journal
ISSN: 2307-4523
https://ijcjournal.org

Project Estimator: A Role-Based Production-Estimation Model for Creative and Game Development Pipelines

Dmytrii Yeremenko*

Art Director, SpinPlay Games, Las Vegas, NV, USA

* *Corresponding author: alens.artist@gmail.com*

Accepted: 28 September 2026 | Published: 24 October 2026

Abstract

Early-stage estimation remains one of the weakest points of creative production and game development. Teams commit to scope, budgets, and deadlines before the structure of the required work has been examined, and a single aggregate figure of hours conceals how the load falls across professional disciplines. This article formalizes a role-based production-estimation model represented as a task-role effort matrix, in which task quantities and per-unit role hours are used to derive role-level workloads, and describes its free, publicly available web implementation, Project Estimator. The model decomposes a project into task types, multiplies each type by its planned quantity, assigns the professional roles that participate in producing 1 unit, and attaches per-unit hours to each role. From these inputs, the model derives the total production volume, workload per discipline and per specialist, projected duration, required team composition, cost at given hourly rates, deadline feasibility, and production bottlenecks. Project duration is determined by the maximum role-level duration under the specified staffing and working-time assumptions, which distinguishes the model from aggregate schedule reasoning that divides total effort by total headcount. The argument follows a review format. The model grew out of more than 20 years of production practice; its spreadsheet predecessor was presented at an industry conference in 2021, and no primary usage data are reported. A worked example with a hypothetical 25-character project illustrates how the role-based view exposes a bottleneck that an aggregate estimate would hide. A structured comparison positions the model against task-management systems and software-estimation methods, and identifies its niche at the stage preceding project launch. Limitations regarding empirical validation and the quality of expert input are discussed, along with directions for future verification.

Keywords: bottleneck analysis; capacity planning; creative pipelines; game development; pre-production; production estimation; role-based model

1. Introduction

Live-service and mobile games have turned visual content into a continuous operational commitment. A product that once shipped as a finished artifact now demands characters, environments, interface layouts, animations, visual effects, seasonal events, and marketing creatives on a fixed cadence, and studios that adopt the service model report big changes in how production work is organized and experienced [1]. Under these conditions, the price of a planning error rises because an underestimated content plan disrupts the release calendar of a live product and pushes teams into sustained overload.

Empirical evidence identifies planning as one of the most painful areas of game production. An analysis of 200 postmortems covering 927 problems found that difficulties tied to management and planning dominate the record of failed expectations across 2 decades of industry practice [2]. A systematic review of digital game project management reports that a large share of game projects proceed without formal project management practices at all, leaving scope, time, and cost decisions to intuition [3]. Game art occupies a distinct position within this picture. The work of game artists combines creative, technical, and communicative competencies, unfolds through situated judgment, and resists reduction to interchangeable task units [4]. One visual element passes through the hands of a concept

artist, a user interface (UI) or user experience (UX) designer, a 2D or 3D artist, an animator, a visual-effects specialist, a technical artist, a developer, and a quality-assurance engineer, and every transition creates a dependency that a total-hours figure cannot capture.

The failure mode addressed in this article follows from that structure. When a project receives a single aggregate estimate, the distribution of load across professional roles remains invisible. The total number of hours may look sufficient while the schedule still collapses, because 1 discipline, a single animator, or a single technical artist, becomes the constraint on everything downstream.

Existing classes of solutions leave this niche open. Task-management systems such as Jira, Asana, Trello, and monday.com support configured work management as their core function, and some platforms and service tiers add workload, capacity, and scenario-planning features. These capabilities operate on work items, projects, or resource allocations that have already been defined in the system, whereas the derivation of discipline-level demand directly from pre-production scope quantities and per-unit role estimates lies outside their primary workflow. Research on software effort estimation offers expert-based practices such as planning poker and story points, along with data-driven and ensemble techniques, yet these approaches predict total effort for software tasks and do not structure workload across creative disciplines [5], [6]. Scaled agile frameworks concentrate on inter-team coordination and organizational mechanisms and stop short of a pre-production estimation model for art pipelines [7], [8].

A further adjacent body of research concerns the resource-constrained project scheduling problem (RCPSP) and its multi-skilled extensions (MS-RCPSP), which model activities, precedence relationships, scarce resources, and skill requirements in order to optimize project duration, cost, resource allocation, or related objectives [9], [10]. These approaches provide richer scheduling and resource-allocation mechanisms once activities, dependencies, and constraints have been defined. The present model addresses an earlier and simpler decision point: converting incomplete creative-production scope into role-level workload, staffing requirements, cost, and deadline feasibility before detailed task scheduling and activity dependencies are available.

To the best of the author's knowledge, the reviewed literature does not identify a model designed for early role-based estimation of creative and game-art production from task quantities and per-unit role effort before detailed scheduling, and hedged phrasing is intentional here, since the absence of such a model in the reviewed literature does not exclude its existence in closed studio documentation.

A conceptual clarification frames the argument. Estimation and capacity planning form 2 sides of 1 question. Estimation asks how much work a chosen scope contains, and capacity planning asks whether a given team can absorb that work within a given interval. Approaches that answer the first question with a single number leave the second question unanswered, since a number of hours says nothing about the professions those hours require. The model presented below binds the 2 questions together at the level of roles, and this binding constitutes its contribution: the same table that measures the scope also tests the team against it.

The article pursues 3 goals. The first goal is to formalize a role-based production-estimation model represented as a task-role effort matrix of task quantities and per-unit role hours. The second goal is to describe Project Estimator, a publicly available, role-based estimation tool that implements the capacity-planning model of the author's broader art-direction methodology. The third goal is to position the model relative to adjacent tool classes and to show that it operates at a stage preceding project launch. The broader methodology itself, which covers organizational architecture, quality gates, and the full content lifecycle, lies outside the scope of this text and appears only where the connection matters.

2. Materials and Methods

The model emerged from production practice. Over more than 20 years in game development and

visual production, including more than 10 years in art-direction roles, the author accumulated estimation tables for planning content scope, staffing, and deadlines across mobile, casino, and live-service projects. A spreadsheet version of the model was presented to the professional community at the Games Gathering conference in 2021, where the estimation table, the balancing of capacity across competencies, and the derivation of team composition were demonstrated on a hypothetical project [11]. The spreadsheet was later converted into a standalone, free, web-based tool, which serves as the reference implementation examined in the article [12].

The formal description of the model is based on 4 primitives. A project contains a set of task types, and character production, background painting, interface screens, mini-games, and promotional materials illustrate the level of decomposition at which the model operates. Each task type receives a planned quantity. Each task type also receives a list of professional roles that participate in producing 1 unit, and each role in that list receives the number of hours required per unit. The product of quantity and per-unit hours, summed across all task types, yields each role's workload. The sum of role workloads yields the total production volume. Projected duration is derived from each role's workload, staffing level, and user-defined working-time capacity, while overall project duration is governed by the role or roles with the longest calculated duration. This rule encodes the central assumption of the model: the calendar length of production is governed by the most-loaded role, and dividing total hours by total headcount misstates the schedule whenever the load is uneven. Cost enters the model through hourly rates attached to roles, and deadline feasibility is determined by comparing the projected timeline with the number of working days available before the target date.

The calculation model can be expressed formally as a task-role effort matrix. Let t denote a task type and r a professional role. For each task type t, q_t denotes the planned quantity, while h_{tr} denotes the number of hours required from role r to produce 1 unit of that task type. If a role does not participate in a particular task type, $h_{tr} = 0$. The total workload assigned to role r, expressed in hours, and the total production effort across all roles are therefore

$$W_r = \Sigma_t(q_t \times h_{tr}), \qquad H = \Sigma_r W_r, \tag{1}$$

where H represents total person-hours. Total effort is not treated as project duration because workloads are distributed unevenly across professional disciplines. Let n_r denote the number of available specialists in role r, and let h_d denote the user-defined number of working hours per person per working day. The tool also accepts a user-defined number of working days per week, denoted by w. The estimated duration of role r in working days and its conversion to calendar days are

$$D_r(work) = W_r / (n_r \times h_d), \qquad D_r(cal) = D_r(work) \times 7 / w. \tag{2}$$

The estimated project duration is determined by the role or roles with the longest calculated duration, and the set of constraining roles, or production bottlenecks, denoted by B, follows from the same expression:

$$D(cal) = \max_r D_r(cal), \qquad B = \arg\max_r D_r(cal). \tag{3}$$

This rule is central to the model: increasing headcount in a non-constraining discipline does not shorten the projected schedule. The overall duration decreases only when the duration of the currently constraining discipline or disciplines is reduced, until another discipline becomes a bottleneck.

When a target deadline L is provided in calendar days, the model converts it into available working capacity and estimates the minimum staffing required for each role. The required number of specialists and the additional staffing recommendation for role r are

$$n_r(req) = \lceil 7W_r / (h_d \times w \times L) \rceil, \qquad a_r = \max(0, n_r(req) - n_r). \tag{4}$$

where a_r represents the number of additional specialists required beyond the currently available team. Deadline feasibility is satisfied when D(cal) ≤ L. If

$D(cal) > L$, the current combination of scope and staffing cannot meet the specified deadline under the assumptions of the model. If c_r denotes the hourly rate assigned to role r, the estimated production cost is calculated as

$$C = \Sigma_r(W_r \times c_r). \quad (5)$$

These calculations are deterministic functions of the task quantities, per-unit role estimates, team composition, working-time settings, hourly rates, and deadline supplied by the user. The model assumes that workload within a discipline can be distributed among available specialists and that specialists within the same role have comparable effective capacity. It does not model coordination overhead, sequential dependencies, vacations, holidays, or differences in individual productivity, and these factors belong to the subsequent scheduling stage described later in the article.

Because the article reports no primary data, the evaluation method is analytical. The model is examined through a structured feature-based comparison with 2 adjacent classes of solutions. The first class contains task-management systems, with Jira, Asana, Trello, and Monday as representative products. The second class covers software estimation methods, with parametric models and expert-based agile practices as representative techniques. The comparison applies 6 criteria: the moment of application relative to project launch, the granularity of workload representation across art disciplines, the capacity to identify bottlenecks by discipline, the logic used to derive duration, support for scenario comparison, and the cost of access, together with the treatment of user data. The criteria mirror the decisions a producer or art director faces during pre-production, keeping the comparison anchored in practice.

The choice of criteria follows the decisions a producer or an art director faces during pre-production. The moment of application separates instruments that shape a plan from instruments that track its execution. Granularity across art disciplines tests whether an instrument sees the profession behind an hour, and bottleneck identification together with the duration logic tests whether it reproduces the mechanism through which uneven load extends a schedule. Scenario comparison tests support for negotiation, and cost with data handling tests accessibility for small studios and independent teams. The comparison covers classes of solutions through representative products and techniques, so the conclusions hold at the level of classes, while individual products add features that shift particular cells over time.

The reference implementation is a browser application. The user enters task types, quantities, roles, per-unit hours, team composition, hourly rates, and a deadline, and the tool recomputes all derived values on every change. The application supports import and export in 2 formats. The comma-separated values (CSV) format preserves the task table with quantities and role hours, and the JavaScript Object Notation (JSON) format preserves the full project structure, including roles, tasks, team, rates, settings, and the deadline. Under ordinary use, the data remain in the browser's local storage, and the user decides what to save, export, or share. A public help center documents the interface and the calculation logic [12].

3. Results

Figure 1 presents the model's structure as a flow from inputs through the calculation core to outputs. The left side of the scheme gathers 3 groups of inputs: the project scope, expressed as tasks with quantities; the per-unit hours assigned to roles; and the constraints, which cover team composition, hourly rates, and the deadline. The calculation core maps task quantities and per-unit role hours into role-level workloads and derives the remaining outputs from those workloads and the project constraints. The right side lists the derived values: workload per discipline, projected duration with the required team, and cost, together with the identified bottlenecks and scenario comparison.



Figure 1: Structure of the role-based production-estimation model

The outputs merit individual attention because each answers a distinct production question: total volume shows how large the project is, workload per discipline shows which professions the scope demands and in what proportion, workload per specialist shows whether the plan respects human limits, projected duration shows when the content can be ready, team composition and cost show whom to hire and what the chosen scope requires in money, deadline feasibility shows whether the target date survives contact with the numbers, and the bottleneck output names the discipline on which the whole schedule hangs. Together, these outputs expose several major dimensions of production feasibility before detailed scheduling begins.

The mechanics of the model become visible in a worked example. The numbers originate from conference material, serve as teaching values, and carry no claim of measurement. For readability, the worked example below expresses effort in person-days. This is equivalent to the hour-based formulation when a fixed number of working hours per day is applied. A hypothetical mobile project requires 25 characters. Production of 1 character involves 4 days of concept art, 10 days of 3D modeling and texturing, 8 days of animation, 5 days of visual effects, and 2 days of technical art, which gives 29 person-days per character and 725 person-days for the full set. An aggregate reading suggests that a team of 5 specialists completes the scope in 145 working days, since 725 divided by 5 equals 145. The role-based reading tells a different story. Figure 2 shows the workload distribution across disciplines for this scope: 3D modeling accumulates 250 person-days, animation 200, visual effects 125, concept art 100, and technical art 50.

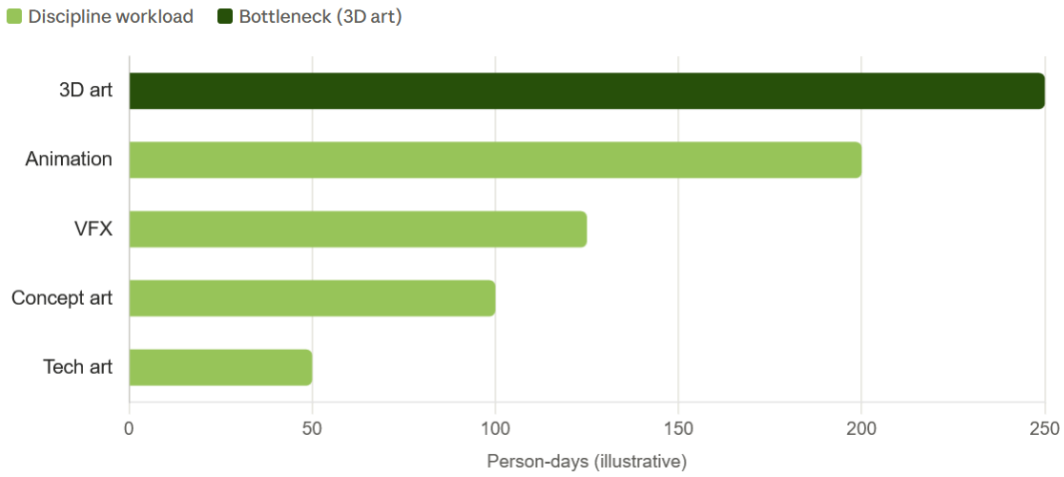


Figure 2: Illustrative workload distribution across disciplines for a hypothetical 25-character project

With 1 specialist per discipline, the 3D discipline governs the timeline at 250 working days, while technical art finishes its 50 days of work 200 days before the end of production and either idles or

requires reassignment. The aggregate figure of 145 days never materializes under this staffing, because no redistribution of hours between an animator and a 3D artist is possible without retraining. Adding a second 3D artist halves the discipline duration to 125 days and moves the animation constraint to 200 days; a further hire in animation shifts the constraint again. Iterative staffing adjustments can move the plan toward a more balanced workload distribution in which per-discipline durations approach one another, and the model makes every step of this adjustment explicit before a single asset enters production.

Scenario comparison extends the same logic to negotiation. When management rejects the staffing that a chosen scope demands, the user reduces quantities and observes the recalculated timeline. When the deadline is immovable, the user increases headcount in the constraining discipline, allocates a share of its load for outsourcing, and checks whether the target date becomes achievable. When hourly rates are entered into the table, every scenario receives a budget, and a conversation about scope turns into a conversation about money, which decision-makers process more quickly.

The model boundary marks the handoff to the next planning stage. Once the estimation table has converged toward a balanced plan, the approved scope, staffing, and timeline are incorporated into a roadmap, where tasks and specialists meet the calendar, including vacations, holidays, dependencies, and integration milestones. The model deliberately stops before that stage, because calendar placement multiplies the number of decisions and buries the structural question of feasibility under scheduling detail. Separating the 2 stages keeps each of them tractable: the estimation table answers whether the project fits the team, and the roadmap answers when each piece of it will happen.

Table 1 summarizes a structured comparison between the model and its implementation and the 2 adjacent classes of solutions across the 6 criteria defined above.

Table 1: Comparison of Project Estimator with adjacent classes of solutions

Criterion	Project Estimator	Task-management systems	Software-estimation methods
Role demand derived directly from pre-production scope quantities	Yes	Not a primary function	Varies
Role-level workload granularity for creative disciplines	Yes	Configuration-dependent	Varies
Role-based bottleneck identification from scope estimates	Yes	Configuration-dependent	Not a primary function
Duration derived from the longest role-level workload under staffing assumptions	Yes	Not a primary function	Varies
Scenario comparison across scope, staffing, rates, and deadline	Yes	Available in some service tiers	Varies
Free public access with local data handling	Yes	No	Varies

The values in Table 1 require explanation. Not a primary function marks capabilities that a class of products does not place at the center of its workflow, even when partial equivalents can be assembled from other features. Configuration-dependent marks capabilities that a diligent team can approximate through custom fields, boards, and workload views once work items exist in the system. Available in some service tiers acknowledges that capacity and scenario-planning functions exist in the premium and enterprise offerings of several platforms, and Varies reflects the breadth of the estimation-method class, which spans parametric models and expert-based agile practices. The comparison does not imply that adjacent task-management platforms lack capacity-planning functionality. It distinguishes the stage at which the calculation begins: Project Estimator derives role demand directly from scope quantities and per-unit role effort before detailed work items and resource allocations have been configured, while the other classes start from work already entered into their systems.

4. Discussion

The main interpretive claim of this article is that the model converts early estimation from an

unstructured guess into an inspectable object. An aggregate estimate such as 725 person-days provides limited visibility into how the workload is distributed across disciplines. A producer who receives a task table with quantities, role hours, per-discipline workloads, and a named bottleneck can challenge any cell, replace it, and watch the consequences propagate. Research on coordination in large development programs shows that shared artifacts such as roadmaps, backlogs, and boards carry a significant share of the coordination load across teams [7]. The estimation table plays the same role at an earlier stage, before any backlog exists, and comparative work on scaled agile methods does not report such early shared representations for creative disciplines, which leaves a gap that the estimation table addresses [8].

The discrepancy between the 2 readings of the worked example follows a general rule. The aggregate reading divides total effort by total headcount, while the role-based reading takes the largest ratio of role workload to role headcount, and the ratio between the 2 durations equals the share of total effort carried by the constraining role multiplied by the total headcount divided by the headcount of that role. In the example, 3D modeling carries 250 of 725 person-days, or 34% of the total, with 1 of 5 specialists, so the role-based duration exceeds the aggregate one by a factor of 1.72, which reproduces the gap between 145 and 250 working days. The same arithmetic explains the idle capacity reported in the results: 5 specialists engaged for 250 working days offer 1,250 person-days, of which 725 carry assigned work, so 42% of paid capacity remains unused under this staffing. After a second 3D artist joins the team, the timeline falls to 200 days and utilization rises from 58% to 60%, since each hire in the constraining discipline shortens the schedule and raises utilization, while a hire in any other discipline adds cost and leaves the duration unchanged. The 2 estimates coincide only when every role carries a workload proportional to its headcount, and any departure from that proportion lengthens the role-based schedule relative to the aggregate figure. The specific values remain illustrative, yet the direction of the effect follows from the structure of the model and holds for any input in which workloads differ across disciplines.

The live-service context raises the stakes of this argument. Case analyses of studios that withdrew from the games-as-a-service model cite strained workers and unmet content expectations as reasons for retreat [13]. A plan that overloads 1 discipline for months belongs to the mechanisms that produce such strain, and a pre-production instrument that exposes the overload before hiring and commitment gives studios a cheap point of intervention. A parallel argument arrives from research on design systems, where a shared source of components and rules improves consistency and reduces friction between design and engineering [14]. Per-unit role norms accumulated in estimation tables function as an analogous shared source for planning: once a studio agrees that 1 character costs 29 person-days in a given style, every subsequent feature negotiation starts from that norm and updates it with experience.

The postmortem analysis by Politowski et al. [2] places management and planning difficulties among the most frequent problems of game production, and the role-based duration rule offers 1 mechanism that can produce such difficulties: a plan whose total hours fit the team still fails when a single discipline carries a disproportionate share of the load. The review by Pieva and Bernardino [3] reports that a large share of game projects proceeds without formal project-management practices, which suggests that an instrument requiring only a task table may meet less resistance in such teams than full scheduling methods. Effort-estimation research concentrates on the accuracy of total effort predictions for software tasks [5], [6], and its techniques remain compatible with the present model, because any of them can supply the per-unit role hours that the task-role effort matrix consumes. Scheduling research on RCPSP and MS-RCPSP treats activities, precedence relationships, skills, and scarce resources in greater detail [9], [10], and in its terms the role-based duration can be interpreted as a capacity lower bound: when the work of each role can be performed only by the specialists of that role, no feasible schedule finishes earlier, and precedence constraints can only extend it. The qualitative study of game art practice by

Hawey [4] describes artistic work as situated judgment that resists reduction to interchangeable task units, which supports the decision to keep roles non-fungible in the model and to treat per-unit hours as expert estimates open to revision.

The limitations of the argument deserve equal attention. First, the model has not been empirically validated in this article, and the numbers in the worked example represent teaching values, so any effect on schedule adherence or cost remains a hypothesis grounded in the author's project experience. Second, the quality of every output depends on the quality of expert per-unit estimates, and a table filled with optimistic hours yields a confident yet wrong plan, a known property of expert judgment in software estimation research [6]. Third, the model abstracts away calendar effects such as vacations, holidays, onboarding, review cycles, and partial parallelism within 1 unit of work; these effects belong to roadmap planning, which follows the estimation stage.

Future work follows from these limitations. The natural next step is an industrial case-study program in which estimates produced with the model are recorded before production and compared with actual outcomes after release, with deviations analyzed per discipline. Anonymized, consent-based usage data from the public implementation would enable a study of how practitioners structure task tables in the wild. A further direction concerns assisted estimation, where accumulated per-unit norms and machine-learning techniques from the effort-estimation literature could propose starting values for role hours that experts then correct [5].

5. Conclusion

This article formalized a role-based production-estimation model for creative and game development pipelines and described its free public implementation. The model rests on a task-role effort matrix of task quantities and per-unit role hours, and derives total volume, per-discipline workloads, duration, team composition, cost, deadline feasibility, and bottlenecks from a single task table. The projected timeline follows from the most loaded discipline, and this rule captures a mechanism of schedule failure that aggregate estimates hide. A worked example illustrated how the role-based view reveals a constraint that turns a nominal 145-day aggregate estimate into a 250-day role-limited schedule under the stated staffing assumptions, and how targeted staffing adjustments can progressively move the workload distribution toward a more balanced plan.

The structured comparison suggests that the model addresses an earlier estimation step than the primary workflow of the adjacent tool classes examined. Task-management systems can support workload and capacity planning once work items and resource allocations have been configured, whereas the present model derives role-level demand directly from pre-production scope quantities and per-unit role estimates. It therefore operates between the initial product brief and detailed production scheduling, where scope, staffing, and deadline assumptions can still be tested at low switching cost. The practical value of the model lies in the transparency of its object: a plan becomes a table that a team, a client, and management can read, dispute, and change together. Verification of the model on industrial cases with recorded estimates and actual outcomes constitutes the main direction of further work, and the public availability of the implementation invites the professional community to participate in that verification.

References

[1] L.-E. Dubois and J. Weststar, “Games-as-a-service: Conflicted identities on the new front-line of video game development,” New Media & Society, vol. 24, no. 10, pp. 2332–2353, 2022, doi: 10.1177/1461444821995815.

[2] C. Politowski, F. Petrillo, G.C. Ullmann, and Y.-G. Guéhéneuc, “Game industry problems: An extensive analysis of the gray literature,” Information and Software Technology, vol. 134, Art. no. 106538, 2021, doi: 10.1016/j.infsof.2021.106538.

[3] L. Pieva and M. Bernardino, “Trends and research in digital game project management: A systematic literature review,” Journal on Interactive Systems, vol. 14, no. 1, pp. 533–545, 2023, doi: 10.5753/jis.2023.3196.

[4] D. Hawey, “Understanding game art practice beyond technical expertise: A qualitative study,” in Proc. 20th Int. Conf. Foundations of Digital Games (FDG ’25), 2025, Art. no. 7, pp. 1–12, doi: 10.1145/3723498.3723724.

[5] B. Alsaadi and K. Saeedi, “Data-driven effort estimation techniques of agile user stories: A systematic literature review,” Artificial Intelligence Review, vol. 55, no. 7, pp. 5485–5516, 2022, doi: 10.1007/s10462-021-10132-x.

[6] J.T.H. de A. Cabral, A.L.I. Oliveira, and F.Q.B. da Silva, “Ensemble effort estimation: An updated and extended systematic literature review,” Journal of Systems and Software, vol. 195, Art. no. 111542, 2023, doi: 10.1016/j.jss.2022.111542.

[7] T. Dingsøyr, F.O. Bjørnson, J. Schrof, and T. Sporsem, “A longitudinal explanatory case study of coordination in a very large development programme: The impact of transitioning from a first- to a second-generation large-scale agile development method,” Empirical Software Engineering, vol. 28, no. 1, Art. no. 1, 2023, doi: 10.1007/s10664-022-10230-6.

[8] H. Edison, X. Wang, and K. Conboy, “Comparing methods for large-scale agile software development: A systematic literature review,” IEEE Transactions on Software Engineering, vol. 48, no. 8, pp. 2709–2731, 2022, doi: 10.1109/TSE.2021.3069039.

[9] S. Hartmann and D. Briskorn, “An updated survey of variants and extensions of the resource-constrained project scheduling problem,” European Journal of Operational Research, vol. 297, no. 1, pp. 1–14, Feb. 2022, doi: 10.1016/j.ejor.2021.05.004.

[10] Z. Bahroun, R. As’ad, M. Tanash, and R. Athamneh, “The multi-skilled resource-constrained project scheduling problem: A systematic review and an exploration of future landscapes,” Management Systems in Production Engineering, vol. 32, no. 1, pp. 108–132, 2024, doi: 10.2478/mspe-2024-0012.

[11] Games Gathering, “Dmitrii Yeremenko – Tvorcheskii protsess dlia ‘fabrik’ po proizvodstvu igr [The creative process for game production ‘factories’],” Internet: https://www.youtube.com/watch?v=eKxTmGOcpZ0, Dec. 1, 2021 [Nov. 1, 2026].

[12] D. Yeremenko, “Project Estimator,” Internet: https://www.projecttoolestimator.online/ [Nov. 1, 2026].

[13] L.-E. Dubois and A. Chalk, “Service withdrawal: The uncertain future of the games-as-a-service model,” Convergence: The International Journal of Research into New Media Technologies, vol. 31, no. 2, pp. 479–499, 2025, doi: 10.1177/13548565241256888.

[14] Y. Lamine and J. Cheng, “Understanding and supporting the design systems practice,” Empirical Software Engineering, vol. 27, no. 6, Art. no. 146, 2022, doi: 10.1007/s10664-022-10181-y.