A financial game with opportunities for fraud

Julian Tritscher*, Anna Krause*, Daniel Schlör*, Fabian Gwinner†, Sebastian von Mammen*, Andreas Hotho*
*Department of Computer Science, University of Würzburg
†Department of Business Management, University of Würzburg
{tritscher, anna.krause, schloer, hotho}@informatik.uni-wuerzburg.de
{fabian.gwinner, sebastian.von.mammen}@uni-wuerzburg.de

***Abstract*—Even though companies store large amounts of business data in enterprise resource planning (ERP) systems, obtaining data for financial fraud detection is prohibitively difficult due to privacy concerns and companies protecting trade secrets. One possible solution is game-based generation of synthetic ERP data, which poses the challenge of designing an environment that generates realistic ERP data and allows players to commit many different types of fraud. In this work, we design a multiplayer game that allows players to cooperatively run a fictional company, while simultaneously challenging them to maximize their personal gain. We introduce an approach for letting players explore fraud scenarios through emergent gameplay and present a prototype that may be primed with information from real world ERP systems to generate realistic data.**

***Index Terms*—Data Generation, Fraud Detection, Machine Learning, Emergent Gameplay, Game Design**

I. Introduction

The Association of Certified Fraud Examiners estimates that companies lose 5% of their revenue to fraud each year [1], demonstrating the potential of using machine learning for automated fraud detection. Although widely used ERP systems contain large amounts of data that can be used to train machine learning algorithms, companies are generally unwilling to release data due to trade secrets and privacy concerns, especially when fraud may be contained in the data [2], [3]. Therefore, existing serious game approaches that imitate regular [4] and fraudulent [5] business processes have been used to generate synthetic data for fraud detection [3]. However, they require players to have in-depth knowledge of complex ERP systems. Following other research disciplines which successfully use games to engage the general public in research-based data generation [6], [7], we aim to develop a financial fraud game that allows non-experts to generate realistic ERP system data for use in machine learning tasks, such as financial fraud detection.

Realistic modelling of fraud is an additional challenge. As fraudsters risk significant damage to their reputation and personal assets, they have an incentive to deliberately create innovative and hard-to-detect frauds, which must be taken into account in game design. Inspired by recent work analyzing and promoting the concept of emergent gameplay, where simple game mechanics lead to complex game scenarios [8], [9], we propose to give players the necessary degrees of freedom to let them explore innovative and well-hidden frauds.

In this work, we design a turn-based multiplayer game, in which players take management positions in a make-to-stock (MTS) production company [10]. We characterize the business process and identify key areas where players may deliberately deviate from regular operation to create complex business frauds. We design the game so that players cooperatively run the company while competitively maximizing their personal profits through good business decisions or well-hidden frauds. To ensure the generation of realistic ERP system data, we propose to use adaptive balancing to adjust the game's statistical models to data from real ERP systems. Finally, we develop a prototype where three players run a cereal production company and demonstrate that our game is capable of generating realistic data of regular business operation.[1]

II. Related Work

Citizen science games let citizens participate in ongoing research projects, with successful applications in biochemistry (Foldit [6]), and quantum research (Quantum Moves [7]) among others. Similar to these games that generate data for their research projects, our game generates data for machine learning based financial fraud detection in ERP systems.

Financial serious games play an important part in education. ERPSim [4] simulates realistic operation of a cereal company and is used to teach players how to interact with the real world SAP ERP system. The White-Collar Hacking Contest [5] pits teams of *hackers* and *detectives* against each other, allowing them to model frauds into existing company data to teach understanding and detection of criminal activities in ERP systems. Here, similar to our proposed game, the aim is not to train fraudsters but momentarily incentivize innovative fraud scenarios for improving economic safety in real life. In contrast to the hacking contest, where the goal is to train participants in fraud detection, our goal is to aid machine learning based detection while also encouraging players to detect fraud attempts from competitors. The aforementioned serious games also all require in-depth player knowledge of ERP systems. Instead, our game models general ERP system functionality in an accessible fashion. We design the game to generate coherent frauds and regular activities simultaneously.

III. Business fraud through emergent gameplay

As explained in Section I, financial fraudsters have an incentive to commit innovative frauds that are difficult to detect. To allow players to create innovative and complex

[1] The prototype is available under https://dmir.org/fraud-game

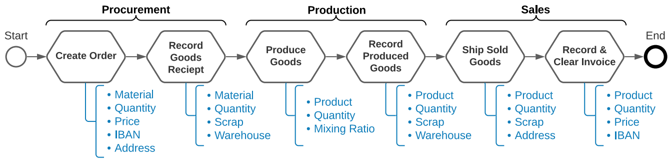


Fig. 1: Linear MTS process with key variables in each step.

fraud scenarios in a game environment, we first identify key variables within existing business processes that may be maliciously altered. In our work, we model fraud cases in a MTS production scenario illustrated in Figure 1.

In the MTS process, the company's procurement orders raw materials from suppliers (*Create Order*) and books them into the company's warehouse (*Record Goods Receipt*). Production takes the goods to the production lines (*Produce Goods*) and books the produced goods back into the warehouse (*Record Produced Goods*). Finally, the sales department ships the goods from the warehouse after purchase (*Ship Sold Goods*) and books the final payments (*Record & Clear Invoice*).

Under each process step, Figure 1 lists multiple key variables we have identified that can be maliciously modified to deviate from the correct business process. Since maliciously changing key variables enables committing fraud (e.g., changing account numbers to steal money from an invoice) or hiding fraud from other players (e.g., recording stolen goods as scrap), we propose to let players freely change key variables (e.g., order prices, shipping details, etc.) and design a game in which the free combination of variable changes allows innovative frauds to emerge.

IV. Game Design Choices

A. Accessible Gameplay

ERP systems of real companies are highly complex and require users with expertise. Inspired by the setting of the ERPSim serious game [4], we focus on making the MTS production process of a cereal production company accessible to large audiences: To allow for strategic play, we organize the game in rounds. Each round represents one business week of the MTS production process, with four weeks forming a month. To reduce complexity and allow players to immerse themselves in the game, we let each month consist of three rounds of *preparation actions*, and one round representing a monthly *business decision*. Preparation actions are used to set up the monthly decision by e.g. establishing and upholding contacts to non-player character (NPC) cooperators and conducting market analysis. In preparation actions, an action point system is used with action points representing the working time which players can allot to preparation actions. This limits the number of actions that can be taken and forces players to strategically plan ahead for their monthly decision. The monthly business decision is used to make all MTS production decisions from Figure 1 for the next month.

B. Game Mode and Player Goals

We design the game as a three-player multiplayer game with asymmetric gameplay that closely models the cooperative and competitive dynamics among employees within a real company, while taking engaging actions for each player from the MTS business process: (1) *Procurement*: purchasing and receiving raw materials, (2) *Production*: planning cereal production through mixing ratios and worker schedules, and (3) *Sales*: choosing a sales market and selling price for final products. Each role may be chosen by one player.

We represent player performance through private bank accounts, issuing two types of bonus payments during gameplay. Firstly, bonus payments are issued to all players when running the company successfully, encouraging players to cooperate. Secondly, players are rewarded for good business strategies through bonus payments that are given based on individual player performance but independent of other players' performance. We also add a competitive element by choosing a winner through the private bank account balance at the end of the game, thereby making the private bank account balance the main game objective. This creates an independent goal for each player to outperform their partners through good gameplay, while also indirectly encouraging fraud as an alternative way to win against better performing opponents.

Conceptually, this game loop has similarities to Public Goods games [11], where multiple players can choose to cooperate by contributing funds to a common pot whose content is increased by a factor and then redistributed equally. Similarly, our game allows players to contribute to company success for shared payoff, whereas players may also choose to not contribute and look for maximizing their own profit. In addition, the individual performance rewards are similar to the incentives for cooperation in the context of Public Goods games.

C. Possibilities for Committing Fraud

Players can commit fraud by deviating from correct business practices. To allow players to commit e.g. theft of goods, we put them in charge of the corresponding business step (here recording and shipping goods) and decouple actually taken actions (goods movements) and the corresponding records in the ERP system (recording goods movements and optionally recording scrap). This allows players to illicitly obtain goods while avoiding discrepancies between real and recorded storage capacities that may be discovered during production.

On a technical level, the game tracks both *real* and *recorded* storage capacities. Players only see the recorded capacities, similar to checking material quantities in an ERP system. Carrying out actions like taking materials out of storage is done on both storage types, causing the game to check whether the requested materials are actually in the real storage. Since our goal is to allow for intentional deviations between real and recorded capacities, we handle divergence only when necessary (e.g. taking materials will fail in case more materials were requested than actually available due to false records). As we track both game states, it enables us to constantly detect

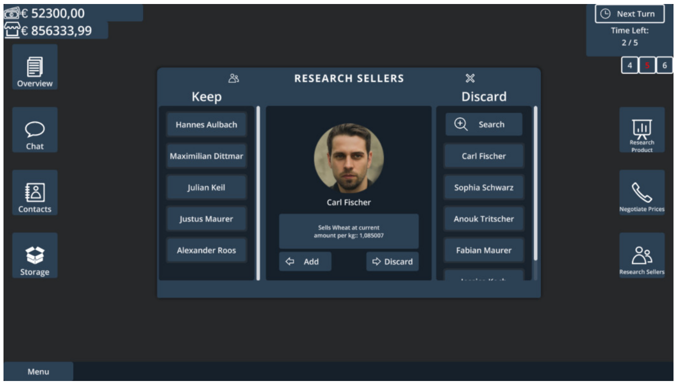


Fig. 2: Weekly preparation action for finding new sales vendors in the purchasing player role.

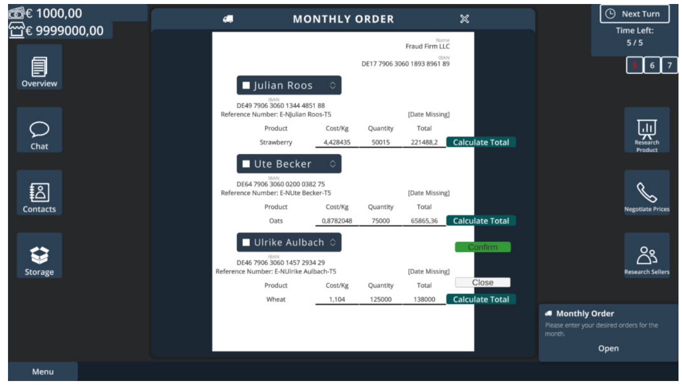


Fig. 3: Monthly business decision for ordering goods in the purchasing player role.

and label divergence as player error or fraud depending on whether a player directly profits from the divergence. This allows us to automatically generate labels for later use in machine learning applications.

To support the creation of realistic, well-hidden frauds, we allow players to take action when they suspect fraudulent behavior among their colleagues. Players may report suspicious behavior from the last round based on available information such as goods in storage or company income and expenses. Successful reports yield bonus payments to incentivize the detection of obvious fraud cases, while unsuccessful reports yield fines to prevent continuous reporting. All players only receive part of the recorded company information, simulating a typical segregation of duties within the company.

V. Game Loop and Prototype

As described in Section IV-A, we implement a turn-based game with a cycle of three preparation turns and one business decision turn, visualized in Table I. Decision turns are offset to prevent waiting times on bought materials, produced goods, or financial assets from sales. Table II shows the possible actions for each player. Players are given 5 action points to spend on preparation actions such as researching NPC vendors shown in Figure 2, with action points enhancing the effect of the chosen preparation action and replenishing after every decision turn.

To produce cereals, the procurement player buys materials from the procurement market (as shown in Figure 3), where generated NPC vendors each offer one type of material with varying prices based on underlying trend functions, material quality, and the relationship with the player. Produced goods are offered on the sales market with player-determined prices and sales regions. Then, NPC distributors choose to buy goods based on product price, regional product preference, product quality and potential connections with the sales player.

As our prototype, we implement the benign business process of the MTS production procedure from Figure 1, and add an explicit type of material theft described in Section IV-C as an exemplary fraud scenario. We give players control over the key variable *Scrap* during the *Record Goods Receipt*, *Record Produced Goods* and *Ship Sold Goods* process steps, allowing them to record materials as broken. For the prototype, we also allow players to directly instruct their NPC contacts to steal and privately sell items from storage. While this is explicitly modeled here to provide a functional prototype, emergent scenarios may be supported in later versions. By letting players take control of the entire flow of goods, players can freely generate fraud cases through creating malicious changes to real and recorded flows of goods within the company, as highlighted in Section IV-C. As counterplay, we implement the fraud reporting system described in Section IV-C. With this prototype, we play out 52 rounds corresponding to a full fiscal year of company operation, before declaring the victorious player based on the highest private bank account balance.

VI. Adaptable Balancing for Data Generation

Since our goal is utilizing data from played sessions for machine learning, modeling realistic data is an important part of game design. Therefore, we design a balancing module that

TABLE I: Monthly turn sequence of player actions where no two decision turns happen concurrently. ☐ denotes preparation turns, ■ denotes decision turns.

Role	Week 1	Week 2	Week 3	Week 4
Procurement	■	☐	☐	☐
Production	☐	■	☐	☐
Sales	☐	☐	■	☐

TABLE II: Preparation actions and business decisions for each player role. Players may choose one action during preparation turns, and perform all business decisions on decision turns.

Role	Preparation actions	Business decisions
Procurement	Find NPC vendors Improve vendor contacts Research material prices	Order materials Record received materials
Production	Find NPC workers Improve worker efficiency	Produce goods Record produced goods
Sales	Find NPC distributors Improve distributor contacts Research product prices	Offer goods for sale Record sold goods

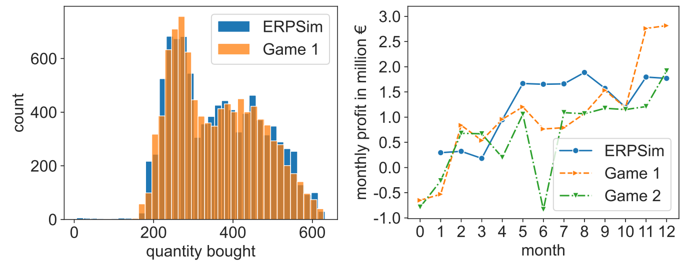
is adaptable to available business data for matching the game scenario to real world data.

Game balancing is customized through adjustable distributions and parameters that can be extracted directly from real ERP system data and set prior to game start. To adjust price trends for individual raw materials and end products, trend functions and duration of trends may be set, while also keeping prices within a given range. This allows us to model price trends similar to observed behavior, while letting gameplay vary for each game run. In the selling market, multiple types of distributors with different buying preferences may be defined. For each type of distributor, buying preferences can be set through multipliers that represent their sensitivity to product quality, regional preference of their customers and relationship to the player. These are multiplied on top of the current market price for a product, yielding the maximum price at which a distributor is willing to buy. The amount of products purchased by distributors is sampled from adjustable random distributions that may be fitted to empirical data. Finally, the monthly production capacity can be adjusted prior to game start.

VII. Demonstration with ERPSim data

To show the prototype's ability to generate realistic data in a player-driven business setting, we adjust the balancing to data from a real SAP ERP system. In this demonstration, we use SAP data generated with the previously mentioned serious game ERPSim [4], since distributions and values of ERPSim data can be published without revealing information relevant to trade secrets and privacy. The ERPSim data was gathered by simulating an entire fiscal year of operation within ERPSim's real SAP ERP system interface. In the game's balancing settings, material and final product prices were modeled with linear price trends and minimum and maximum values taken from the ERPSim data. Production quantities were adjusted to ERPSim's maximum production capacity. For the selling market, three types of distributors were chosen with buy-quantities modeled through approximating ERPSim data with multivariate Gaussian distributions. The number of generated distributors was calculated to match the total quantities of purchased goods in ERPSim.

For demonstration purposes, we let two three-player student teams play online sessions of 52 in-game weeks, causing total sessions to last about 90 minutes of playtime. We restrict players to non-fraudulent behavior to showcase the game's similarity to and potential divergences from the underlying ERPSim data. In Figure 4a we extract and visualize the quantities of sold cereals for all sales from ERPSim and our prototype (Game 1) during the recorded year. The adaptable balancing scheme from Section VI matches the distribution of the given ERPSim data well, successfully mimicking both the quantities per sale and the total sales of ERPSim. In combination with the cereal prices being in similar ranges to ERPSim data with linear price trend variations, this amounts to €$23.7M$ of positive income per year, in comparison to ERPSim's €$19.5M$. In Figure 4b we display the raw monthly profit achieved, calculated by subtracting the amounts spent on materials from the positive income gained through sold goods. While the ERPSim data is offset by one month since the turn-based game requires one full month to fully start production, the trend of our Game 1 & 2 runs closely matches the profits achieved in ERPSim for most months. We also note that player choices during monthly business decisions allow teams to perform poorly in the economic setting, providing challenging gameplay while yielding ERPSim-like data on runs where players make good business decisions.



(a) Distribution of cereal quantities bought per purchase

(b) Monthly profit for ERPSim and multiple game runs

Fig. 4: Plots showing similarity of ERPSim generated data and game runs from the adjusted prototype.

VIII. Conclusion

In this work we presented a turn-based three player cooperative multiplayer-game that encourages players to make good business decisions while giving an indirect incentive for committing well-hidden financial frauds. We identified key variables in a MTS production scenario that may be changed to commit or hide fraud and argued that giving players control over these variables may cause fraudulent activities through emergent gameplay. We presented a prototype and showed that we are capable of adapting its balancing to mimic realistic ERP system data, showing that the game generates promising data for training machine learning approaches.

Overall, we have laid the foundation to utilize emergent gameplay for creating well-hidden fraud scenarios within ERP system data. However the degrees of freedom given to the players are still limited in the current prototype. Our next steps are therefore to model the prototype's theft-fraud scenario less explicitly by giving players direct control over the flow of goods, to introduce further degrees of freedom by giving players control over additional key variables from Figure 1, and to apply the generated data in machine learning-based fraud detection. Further, we aim to conduct user studies to ensure that gameplay is enjoyable for all roles, thus encouraging citizen participation.

Acknowledgement

The authors acknowledge the financial support by the Federal Ministry of Education and Research of Germany as part of the DeepScan project (01IS18045A). The prototype was developed in a student project by Hannes Aulbach, Maximilian Dittmar, Julian Keil, Justus Maurer, and Alexander Roos.

References

[1] ACFE, "2020 Global Occupational Fraud Study," *Report To the Nations*, [Online; accessed 28. May. 2021]. [Online]. Available: https://www.acfe.com/report-to-the-nations/2020

[2] Y. Yannikos, F. Franke, C. Winter, and M. Schneider, "3lspg: Forensic tool evaluation by three layer stochastic process-based generation of data," in *Int. Workshop on Computational Forensics*. Springer, 2010.

[3] G. Baader and H. Krcmar, "Reducing false positives in fraud detection: Combining the red flag approach with process mining," *Int. Journal of Accounting Information Systems*, 2018.

[4] P. Léger, J. Robert, G. Babin, R. Pellerin, and B. Wagner, "Erpsim," *ERPsim Lab (erpsim. hec. ca), HEC Montreal, Montreal, Qc*, 2007.

[5] M. Schermann and S. R. Boss, "The white-collar hacking contest: A novel approach to teach forensic investigations in a digital world," in *Dewald Roode Workshop on Information Systems Security Research*, 2014.

[6] S. Cooper, F. Khatib, A. Treuille, J. Barbero, J. Lee, M. Beenen, A. Leaver-Fay, D. Baker, Z. Popović *et al.*, "Predicting protein structures with a multiplayer online game," *Nature*, 2010.

[7] J. J. W. Sørensen, M. K. Pedersen, M. Munch, P. Haikka, J. H. Jensen, T. Planke, M. G. Andreasen, M. Gajdacz, K. Mølmer, A. Lieberoth *et al.*, "Exploring the quantum speed limit with computer games," *Nature*, 2016.

[8] C. Ampatzidou, "Reinventing the rules: Emergent gameplay for civic learning," in *The Hackable City*. Springer, Singapore, 2019.

[9] S. Wodarczyk and S. von Mammen, "Emergent multiplayer games," in *2020 IEEE Conference on Games (CoG)*. IEEE, 2020.

[10] S. Rajagopalan, "Make to order or make to stock: model and application," *Management Science*, 2002.

[11] A. Chaudhuri, "Sustaining cooperation in laboratory public goods experiments: a selective survey of the literature," *Experimental economics*, vol. 14, no. 1, 2011.