# Model-Driven Game Development: A Summary of a Mapping Study

**Lucas Martins**[1], **Fábio Paulo Basso**[1], **Elder de Macedo Rodrigues**[1], **Maicon Bernardino**[1]

[1]Laboratory of Empirical Studies in Software Engineering (LESSE), Federal University of Pampa (UNIPAMPA), Alegrete, RS, Brazil

lgatts@gmail.com,

{fabiobasso,elderrodrigues}@unipampa.edu.br, bernardino@acm.org

***Abstract.*** *Since 2014, it has been estimated that the digital gaming industry would outperform the film industry as a matter of profit within the entertainment industry, which has already exceeded at least twice as much sales as music. In order to gain productivity in the development of cross-platform games, research efforts are directed towards some tasks automation through Domain Specific Languages (DSL) and Model-Driven Game Development (MDGD) approaches scoping code generation or runtime execution. The present work is a systematic literature mapping study that aims to analyze the current research effort in MDGD, evaluating them in the context of the technology transfer to the industry. By means of six research questions, we conclude that the technology transfer to the industry is little reported and it is still carried out ad hoc.*

***Resumo.*** *Desde 2014, já se estimava que a indústria de jogos digitais superaria a indústria de filmes na questão de lucro dentro da indústria do entretenimento, tendo esta já ultrapassado em no mínimo o dobro de vendas que a de música. De modo à obter produtividade no desenvolvimento de jogos cross-plataformas, esforços de pesquisa são dirigidos para automatização de tarefas de desenvolvimento por meio de Linguagens Específicas de Domínio (DSL) e desenvolvimento de jogos dirigidos por modelos (MDGD, do inglês Model-Domain Game Development) usando-se de geração de código ou uso de modelos em tempo de execução. O presente trabalho é um estudo de mapeamento sistemático de literatura que visa analisar o panorama atual das pesquisas em MDGD, avaliando-as no contexto da transferência de tecnologia para a indústria. Por meio de seis questões de pesquisa, concluímos que a transferência dessas tecnologias para a indústria é pouco relatada e ou ainda é realizada de modo ad-hoc.*

## 1. Introdução

Desde 2014, já se estimava que a indústria de jogos digitais superaria a indústria de filmes na questão de lucro dentro da indústria do entretenimento, tendo esta já ultrapassado em no mínimo o dobro de vendas que a de música (Fluery et al. 2014). Concomitantemente com este crescimento no mercado, vê-se necessária a melhoria nos processos de desenvolvimento (Whittle et al. 2015) também dos jogos digitais. Como relatado em (Kasurinen et al. 2017), em muitos casos, jogos digitais são feitos de forma

*ad-hoc* por pessoas que desenvolvem por hobby, ou pequenas equipes multidisciplinares de desenvolvedores independentes (*indies*) e artistas que não seguem nenhum tipo de processo documentado. O mesmo cenário é observado na indústria de grande porte (Kasurinen et al. 2017).

Olhando para o desenvolvimento de jogos digitais e para a Engenharia de Software, em (Kasurinen et al. 2017), conclui-se que a sua combinação na prática apresenta enormes desafios. Um deles é quanto à produtividade, já que por algum tempo as empresas do setor vem buscando a conciliação de disciplinas da Engenharia de Software (Petrillo et al. 2009), mas sem sucesso quanto se trata de maior produtividade (Kasurinen et al. 2017). Portanto, por mais que um dos objetivos das empresas de jogos digitais seja o de permitir um maior investimento de tempo no processo criativo e na fase de testes, o que garantiria a entrega de um produto final original e com maior qualidade, o requisito de produtividade muitas vezes supera as necessidades da garantia da qualidade (Kasurinen et al. 2017).

Prometendo maior produtividade com qualidade, algumas pesquisas vêm colocando esforços na automatização de desenvolvimento de games por meio de Linguagens de Domínio Específico (DSLs) e abordagens para desenvolvimento de jogos dirigidos por modelos (MDGD - *Model-Driven Game Development*) (Matallaoui et al. 2015; Prado and Lucrédio 2015; Zhu et al. 2016). Neste contexto, nossa contribuição é um relato resumido de um mapeamento sistemático de literatura, analisando e elencando as DSLs propostas para abordagens de MDGD e caracterizando-as no estado da prática. Em especial, buscou-se encontrar as propostas que, de algum modo, permitem a execução de um jogo utilizando modelos como fonte de informação. Ou seja, modelos usados em tempo de execução, no que é conhecido por Models@runtime (Blair et al. 2009). Por fim, analisamos se e como as propostas para MDGD são transferidas para a indústria de jogos digitais, elencando os principais lacunas de pesquisa da área para a prática de desenvolvimento de jogos digitais.

O presente estudo é organizado como segue: a Seção 2 traz alguns conceitos importantes que contextualizam este estudo de mapeamento sistemático, o qual é descrito na Seção 3. A Seção 4 apresenta os resultados encontrados na revisão de literatura, cujas ameaças à validade são discutidas na Seção 5. Por fim, na Seção 6 se faz reflexões sobre o estado atual e as possibilidades de trabalhos futuros em MDGD.

## 2. Embasamento Teórico

Para se iniciar a discussão sobre desenvolvimento dirigido por modelos no desenvolvimento de jogos, alguns conceitos, tanto sobre engenharia de modelos quanto sobre o desenvolvimento de jogos, devem ser expostos.

### 2.1. Ferramentas utilizadas no desenvolvimento de jogos

Apesar de muitos esforços na criação de DSLs (Domain Specific Languages) para o desenvolvimento de jogos (Sánchez et al. 2015; Suzuki et al. 2011; Suzuki et al. 2011; Carette 2011), alguns autores (Prado and Lucrédio 2015; Guo et al. 2015a; Purushothaman et al. 2017; Furtado and Santos 2006) consideram na atualidade três tipos de ferramentas:

- **APIs e Bibliotecas:** São utilizadas por abstrair interações diretas com o hardware no processamento e renderização de imagens, som, entradas como joystick, mouse e teclado. Assim, permitindo uma maior agilidade no processo de desenvolvimento de jogos.
- **Visuais:** São ferramentas que permitem para programadores experientes a rápida prototipação de jogos para poder vender suas ideias. Podem ser usadas sem nenhum tipo de conhecimento em programação, apenas lógica, podendo ser utilizadas por artistas e hobbistas. Porém, geralmente possuem algum tipo de linguagem para se criar *scripts* e permitir alguma flexibilidade. Alguns exemplos desse tipo de ferramentas são Game Maker, Stencyl, RPG Maker, Construct 2.
- **Engines:** Segundo (Zhu et al. 2016) podem ser consideras o estado da arte em Engenharia de Software no desenvolvimento de jogos, além de utilizarem das APIs e bibliotecas mencionadas anteriormente. Ainda, possibilitam a criação de certos processos visuais por meio de ferramentas de *drag and drop*. Também auxiliam na manipulação de animações de forma mais conveniente e possibilitam a integração com ambientes de programação e linguagens de programação, permitindo a criação de jogos mais complexos. Além disso, permitem a depuração e algum tipo de automatização de testes.
- **Os engines mais usados:** As grandes indústrias geralmente utilizam *engines* proprietárias. Porém, existem engines de uso gerais utilizadas por equipes menores e até algumas maiores (Kasurinen et al. 2017). Dois exemplos de *engines* mais utilizadas são Unreal e Unity, cada qual com suporte de DSLs internas que popularizaram a sua adoção.

## 2.2. Desenvolvimento dirigido por modelos (MDD)

A principal motivação do MDD é a possibilidade de desenvolver software em maior alto nível, não se preocupando com detalhes de implementação e reutilização e geração automática de código, o que muitas vezes garante uma maior confiabilidade no código gerado, uma vez que se é reutilizado, é muito provável que já tenha sido testado (Sommerville 2010).

Na área de jogos, existe a motivação de permitir profissionais menos familiarizados com programação, como artistas, poderem desenvolver, se preocupando mais com a criatividade e inovação ao invés de detalhes de linguagem, plataforma etc. Este é um bom pretexto para o uso do MDD, simplificando o desenvolvimento para não desenvolvedores por meio de DSLs que aumentam o nível de abstração. Já em indústrias de jogos maiores, que geralmente desenvolvem jogos AAA (TripleA), que observa-se maior risco financeiro quando comparados a *blockbusters* na indústria cinematográfica, o MDD pode não ser tão interessante uma vez que especialistas em programação para determinadas plataformas são abundantes.

Zhu e colegas afirmam que abordagens para MDD possuem as seguintes características que poderiam influenciar positivamente para adoção na indústria de games: 1) suporte para maior complexidade e tipos de interações mais amplos do que é possível se obter em curto tempo com abordagens *code-first* (Zhu 2014); 2) a automatização de partes repetitivas se torna essencial e, através da geração de código por modelos, isso pode ser atingido, deixando para os desenvolvedores mais tempo para focar em funcionalidades mais complexas do sistema (Zhu et al. 2016).

## 2.3. Models@Runtime

Um tipo particular de trabalhos que utilizam modelos como elementos de primeira ordem no desenvolvimento de aplicativos é caracterizado como Models@Runtime. Este tipo

de trabalho busca embutir nos modelos elementos que possam ser usados para sistemas adaptativos (Cheng et al. 2014), os quais possam ser reconfigurados em tempo de execução. Segundo (Blair et al. 2009), o uso de modelos em tempo de execução apresenta dois benefícios que abordagens de MDD não apresentam: 1) permite uma rápida validação do que foi modelado, uma vez que não há a necessidade de geração de código, e; 2) reduz o tempo para produzir aplicações. Em (Oliveira et al. 2018) os autores mencionam como um benefício em relação à geração de código o Retorno de Investimento (ROI).

## 3. Metodologia

Esta seção apresenta um resumo do protocolo do estudo. Visando abranger a maior quantidade possível de trabalhos, um mapeamento sistemático de literatura provê uma visão ampla da linha de pesquisa com base em uma ou mais questões formuladas na execução da mesma. Assim, garante-se uma maior confiança nos resultados, reduzindo a possibilidade de parcialidade devido a hipóteses de pesquisa.

O protocolo foi realizado de acordo o *guideline* (Kitchenham and Charters 2007) e segue a seguinte ordem lógica: 1) foi definido os objetivos e questões de pesquisa; 2) então foram definidas as bases nas quais se realizaria a pesquisa; 3) depois foi elaborada uma *string* de busca geral e adaptada para a sintaxe de cada base, bem como questões de pesquisa, critérios, de inclusão e exclusão; 4) em seguida, após realização da busca a aplicação dos critérios por título, *abstract* e *keywords* foram filtrados os trabalhos relevantes; e 5) Por fim, os trabalhos que permaneceram foram lidos por completo e desses foram extraídas as informações que serão apresentadas na Seção 4.

### 3.1. Objetivo e questões de pesquisa

O principal objetivo do trabalho foi mapear o estado da arte em pesquisas envolvendo *Model Driven Game Development* (MDGD) a partir do ano de 2012. Este ano é importante porque marca o início da oitava geração de *consoles* e, consequentemente, o aumento na complexidade dos jogos que começam a ser produzidos aproveitando o máximo de capacidade desses *consoles*. Ou seja, 2012 é considerado um marco com o lançamento do Nintendo Wii U, após em 2013 surgem os consoles Playstation 4 e Xbox One. Além disso, caracteriza o avanço de processamento nos computadores pessoais e *mobile*, impulsionando o uso de abordagens para geração de código para múltiplas plataformas. Com base nisso, foram elaboradas as seguintes questões de pesquisa:

- Q1. Quais são as principais abordagens utilizadas nas pesquisas que propõem a utilização de MDE para desenvolvimento de jogos? Nosso objetivo é caracterizar estes estudos como provedores de tecnologia para MDGD, o que requer a identificação de sua natureza técnica (DSL's Textuais, Gráficas, Meta-Modelos, etc).
- Q2. As pesquisas consideram a utilização de MDE em conjunto com *Game Engines*? Nosso objetivo é caracterizar os estudos que se ligam com ferramentas externas usadas atualmente na indústria de jogos digitais.
- Q3. As pesquisas têm algum tipo de validação em casos de uso reais da indústria? Para as pesquisas que relatam a transferência de tecnologia para a indústria, nosso objetivo é principalmente identificar os fatores de produção associados com os recursos disponíveis em suas propostas.
- Q4. As pesquisas levam em consideração a complexidade devido a variedade de gêneros de jogos digitais existentes? Nosso objetivo é identificar os contextos os quais os estudos foram aplicados.
- Q5. As pesquisas levam em consideração as diferentes arquiteturas de jogos? Nosso objetivo é identificar a aplicabilidade dos recursos para diferentes plataformas (*Standalone*, *Massive Multiplayer*, Cliente-Servidor, P2P, etc).
- Q6. As pesquisas suportam algum modelo em tempo de execução (Models@runtime)? Uma vez que modelos podem ser utilizados em tempo de execução, nosso objetivo é identificar se, de algum modo, as pesquisas para MDGD estão adotando esta abordagem de desenvolvimento de software.

### 3.2. Resumo do Protocolo

Foi decidido utilizar cinco bases para a pesquisas: Scopus, IEEE xplore, Science Direct, ACM digital library e SpringerLink. A *string* geral de busca é representada como segue:

| (“Game Development”) AND (“Model-Driven” OR “Model-Based” OR “MDD” OR “MDGD” OR “MDE” OR “MBE”) |
|---|

Os critérios de exclusão adotados são:

- CE1. Data de publicação igual ou acima de 2012.
- CE2. O artigo não trata sobre o desenvolvimento de games.
- CE3. O artigo não está escrito em inglês.
- CE4. Artigo não disponível para download pelo periódicos da CAPES ou não tem compartilhamento gratuito na web.

## 4. Resultados e Discussão

Como se pode observar a partir do referencial teórico, umas das principais dificuldades de se solidificar o uso de DSL no desenvolvimento de jogos, apesar dos esforços já realizados em estudos sobre isso, é a grande variedade de gêneros e arquiteturas. Portanto, nessa seção se faz uma análise sobre os estudos encontrados e qual suas respostas para as questões de pesquisa.

### 4.1. Estudos Selecionados

Após juntar os resultados de todas as pesquisas nas bases selecionadas, foram obtidas um total de 110 referências. Sobre essas, foi realizando o primeiro critério de exclusão que era a data de publicação igual ou acima de 2012, removendo 45 referências. Para as restantes, aplicou-se os critérios CE2 e CE3 pela leitura de todos os *abstracts*. A aplicação do último critério de exclusão resultou em apenas 11 artigos, apresentados na Tabela 1.

**Tabela 1. Trabalhos finais selecionados durante o mapeamento sistemático**

| Id | Referência | Título | Ano |
|---|---|---|---|
| S01 | (Dormans 2012) | The Effectiveness and Efficiency of Model Driven Game Design | 2012 |
| S02 | (Fernandez et al. 2012) | Integrating Usability Evaluation into Model-Driven Video Game Development | 2012 |
| S03 | (Prado and Lucrédio 2015) | A Flexible Model-Driven Game Development Approach | 2015 |
| S04 | (Guo et al. 2015a) | A Workflow for Model Driven Game Development | 2015 |
| S05 | (Guana et al. 2015) | Building a Game Engine: A Tale of Modern Model-Driven Engineering | 2015 |
| S06 | (Matallaoui et al. 2015) | Model-Driven Serious Game Development: Integration of the Gamification Modeling Language GaML with Unity | 2015 |
| S07 | (Guo et al. 2015b) | RealCoins: A Case Study of Enhanced Model Driven Development for Pervasive Games | 2015 |
| S08 | (Céspedes-Hernández et al. 2015) | SEGA-ARM: A Metamodel for the Design of Serious Games to Support Auditory Rehabilitation | 2015 |
| S09 | (Zhu et al. 2016) | Engine- Cooperative Game Modeling (ECGM): Bridge Model-Driven Game Development and Game Engine Tool-chains | 2016 |
| S10 | (Purushothaman et al. 2017) | Automation approach for cocos-2dx based multi-player cardgame for web and mobile | 2017 |
| S11 | (Guo et al. 2018) | Ontology-Based Domain Analysis for Model Driven Pervasive Game Development | 2018 |

### 4.2. Respostas às Questões de Pesquisa

Com base nesses artigos selecionados foram respondidas as questões de pesquisa:

**Q1. Quais são as principais abordagens utilizadas nas pesquisas que propõem a utilização de MDE para desenvolvimento de jogos (DSL's Textuais, Gráficas, Meta-Modelos, etc)?**

Nota-se que existe uma predominância entre o uso de linguagens textuais {S04, S07, S10, e S11} e algumas misturas com representação visual. Isso pode ocorrer devido ao contexto dos desenvolvedores, familiares com linguagens de programação. A simplicidade do desenvolvimento de uma linguagem textual também deve ser considerada, assim como o fácil acesso à ferramentas que possibilitam esse desenvolvimento como xText[1]. Também pelos gêneros de jogos, que serão abordados na questão 4, não serem tão abrangentes. Por exemplo, devido à complexidade no desenvolvimento, os trabalhos {S03, S09} necessitam tratar de mais de um gênero com o uso de mais de uma DSL. Assim, um ponto em aberto nos estudos é a falta de critérios de qualidade de DSLs por gênero de game.

**Q2. As pesquisas consideram a utilização de MDE em conjunto com *Game Engines*?**

Dos artigos, apenas 4 estudos {S06, S03, S09, e S10} desenvolveram a própria *engine* como um nível de abstração maior e independente de plataforma. Assim, utilizando MDE, demonstram como transformar um modelo independente de plataforma noutro dependente de uma *engine* para jogos. Também percebe-se que as engines são otimizadas continuamente para configurações de hardware/GPUs, apresentando contextos heterogêneos para desenvolvimento, o que requer reprogramação. Neste sentido, elevar o nível de abstração em MDD pode ser benéfico para este cenário. Uma vez que as *engines*/plataformas de jogos tendem a evoluir cada vez mais dentro da indústria dos jogos, concluímos que a abstração independente de plataformas no desenvolvimento de jogos das engines de execução é pouco explorada, e, portanto, caracteriza um espaço interessante para explorar como um meio para facilitar a transferência de tecnologia de MDGD.

**Q3. As pesquisas têm algum tipo de validação em casos de uso reais da indústria?**

Nenhum dos trabalhos possuem validação com jogos comerciais. Já que há um risco associado com a transferência desse tipo de tecnologia na indústria (Whittle et al. 2015), entendemos que há uma barreira de entrada para estas ferramentas na indústria. Muito provavelmente, a falta de trabalhos discutindo como as tecnologias são recebidas em ambientes reais de desenvolvimento de software se deve à dificuldade de se encontrar empresas dispostas a apostar na utilização de novas ferramentas e metodologias para MDGD. Assim, concluímos que para fomentar a credibilidade para a introdução do MDGD em contextos de Fábricas de Software (Basso et al. 2017), seria necessário a utilização de relatos sobre a eficiência de MDGD como um grande desafio.

**Q4. As pesquisas levam em consideração a complexidade devido à variedade de gêneros de jogos digitais existentes?**

Apenas dois trabalhos {S03 e S09} propõem o desenvolvimento de DSL que atenda de forma mais genérica, independentemente do gênero. Os demais trabalhos são focados em gêneros específicos, como *serious games* {S01, S06 e S08} e *pervasive games*

[1]https://www.eclipse.org/Xtext/

(Todos do mesmo grupo de autores {S04,S07 e S11}). Essa limitação torna a visibilidade e aplicabilidade das DSLs apresentadas menos aplicáveis no cenário atual das empresas de jogos digitais. Assim, um mapeamento das necessidades para MDGE para uma diversidade maior de gêneros caracteriza um ponto interessante para pesquisas na área. Nesta direção, os trabalhos {S03 e S09} argumentam que, para o desenvolvimento de jogos independentemente do gênero, faz-se necessária a integração de várias DSLs, o que pode tornar a transferência dessa tecnologia para contextos específicos bem complexa.

**Q5. As pesquisas levam em consideração as diferentes arquiteturas de jogos (*Standalone*, *Massive Multiplayer*, Cliente-Servidor, P2P, etc)?**

Apenas um trabalho {S09} aborda diferentes arquiteturas no suporte para MDGD. Esse pequeno número evidencia da falta de maturidade da área de pesquisa para a transferência de tecnologia de MDE. Ou seja, faltam estudos de caracterização das características estruturais para o desenvolvimento de arquiteturas de jogos digitais e DSLs. Assim, um bom tópico de pesquisa à ser explorado como próximo trabalho é o mapeamento de arquiteturas dos jogos independentemente de *model-driven*. Portanto, somente munido de tais características estruturais é possível fazer um comparativo da cobertura das DSLs no suporte para MDGD.

**Q6. As pesquisas suportam algum modelo em tempo de execução (Models@runtime)?**

Não foram encontradas abordagens para MDGD que utilizem modelos em tempo de execução. Ou seja, todas as propostas incluídas tratam da geração de código baseada em modelos. Portanto, esta limitação na literatura da área caracteriza uma boa oportunidade de pesquisa.

## 5. Ameaças à Validade

Este estudo pode sofrer algumas ameaças à validade, as quais alcançam validades internas, do construtor e da conclusão do estudo como segue:

**Validade do Construtor:** Adotamos cuidadosamente procedimentos para recuperar e filtrar os estudos primários. Palavras-chave e seus sinônimos foram definidos de acordo com métodos bem estabelecidos em (Kitchenham and Charters 2007). Uma ameaça não tratada ao constructo é o pequeno número de artigos selecionados. Assim, pretende-se executar uma revisão complementar do tipo *snowballing*.

**Validade Interna:** Esse tipo de ameaça refere-se à validade da análise realizada. Isto é, se as conclusões derivadas dos dados são internamente válidas. Cada estudo primário identificado implementa sua própria abordagem para *design*. Entender e mapear cada técnica pode ser visto por si só como uma ameaça à validade. Com isso em mente, procuramos mitigar esse risco discutindo e analisando de perto todos os autores do artigo. Além disso, uma preocupação sempre presente durante todo o estudo era garantir que os estudos primários selecionados fossem consistentemente identificados e analisados. Para isso, investimos cuidadosamente um enorme esforço para filtrar os estudos primários.

**Validade da Conclusão:** Essas ameaças estão estritamente relacionadas a problemas que podem afetar a confiabilidade de nossas conclusões. Primeiro, seguimos rigorosamente as etapas fornecidas pelo protocolo de estudo de mapeamento sistemático bem difundido (Kitchenham and Charters 2007). Finalmente, todas as conclusões

deste artigo foram feitas após a coleta dos resultados, evitando a taxa de erro e de *fishing* (Wohlin et al. 2012).

## 6. Conclusões e Trabalhos Futuros

Este trabalho apresentou um mapeamento sistemático para identificar como a transferência de tecnologia existente para o suporte de *Model-Driven Game Development* (MDGD) ocorre na indústria. Para tal, o mapeamento considerou seis questões de pesquisa aplicadas sobre 11 artigos selecionados. Concluímos que a área de pesquisa em MDGD é embrionária, com sua aplicação ainda em cenários acadêmicos. Existe, portanto, muito espaço para a realização de novas pesquisas.

Uma das principais dificuldades na adoção de DSLs para jogos é justamente a grande quantidade de gêneros de jogos e arquiteturas. Ou seja, é preciso um grande número de DSLs para cada gênero. Os gêneros variam de jogos de texto contando histórias interativas, *massive multiplayer online role-playing game* (MMORPGs), jogos de tiro em primeira pessoa (FPS) e muitos outros, os quais não possuem muitas características em comum. Em termos de arquitetura, jogos podem ser *standalone*, rodando apenas na máquina do usuário, podem possuir servidores onlines onde armazenam todas as informações do jogador, podem possuir servidores apenas para guardarem informações como *ranking* e ainda podem ser P2P onde cada computador é cliente e servidor ao mesmo tempo em uma versão *multiplayer*. Nesse sentido, os 11 trabalhos analisados cobrem apenas uma fração pequena desta variedade.

Relatos do estado da prática em ferramentas diversas para *model-driven* são abundantes na literatura (Whittle et al. 2015; Selic 2014). No entanto, pouco discute-se em relação aos problemas existentes nas práticas para a transferência de tecnologia. Encontramos que este também é o caso dos 11 artigos analisados. Assim, já que o conjunto de artigos selecionados não demonstraram aplicação na indústria, não foi possível compreender como o MDGD é usado na prática. Outros trabalhos deveriam, portanto, buscar esclarecimentos de aspectos importantes que não temos as respostas como: as barreiras de entrada do mercado, os benefícios e problemas associados com essas ferramentas, e se benefícios como aumento de produtividade em sistemas de informação web (Oliveira et al. 2018) podem ser obtidos também para MDGD por meio de abordagens para Models@Runtime.

Em termos de transferência de tecnologia, entendemos porque um dos principais desafios é conseguir levar esses estudos de MDGD para dentro da indústria como forma de comprovar sua eficiência. Ou seja, falta convencimento empírico associado com abordagens para MDGD. Assim, é preciso conduzir estudos de caso reais, os quais dêem credibilidade e fomentem a literatura com dados voltados para a tomada de decisão estratégica nas empresas.

Também observa-se que integração das DSLs com o ambiente de desenvolvimento já utilizado na indústria (Zhu et al. 2016), que são as *Game Engines*, é algo desejável para a transferência de tecnologia MDGE. Isso nos remete num aprendizado crucial para o sucesso na transferência de tecnologia de MDE: adapte as ferramentas às pessoas e não o contrário (Whittle et al. 2015). Portanto, abordagens para transferência de tecnologia por meio de integração, como a MDE as a Service (Basso et al. 2017), podem ser exploradas também no contexto de MDGD.

Por fim, para angariar mais estudos publicados no tema, um trabalho futuro

pretende realizar a execução de um estudo complementar via protocolo *snowballing*. Em especial, buscam-se contribuições para MDGD seguindo abordagens para Models@Runtime, que em nossa opinião caracteriza uma grande lacuna na área investigada. Assim, novos trabalhos poderiam explorar atributos de MDE como adaptação ao contexto dos jogadores, que pode se dar por meio de linha de produto dinâmica ou reconfiguração baseadas em sensores e atuadores de dispositivos, como de captura de expressões faciais para aumentar a imersão, em abordagens de reconfiguração em tempo de execução.

## Referências


[Basso et al. 2017] Basso, F. P., Oliveira, T. C., Werner, C. M., and Becker, L. B. (2017). Building the foundations for 'mde as service'. *IET Software*, 11:195–206(11).

[Blair et al. 2009] Blair, G., Bencomo, N., and France, R. B. (2009). Models@ run.time. *Computer*, 42(10):22–27.

[Carette 2011] Carette, L. B. J. (2011). Saga: A dsl for story management. In *DSL 2011: International Conference on Domain-Specific Languages*, pages 1–20.

[Céspedes-Hernández et al. 2015] Céspedes-Hernández, D., Pérez-Medina, J. L., González-Calleros, J. M., Álvarez Rodríguez, F. J., and noz Arteaga, J. M. (2015). Sega-arm: A metamodel for the design of serious games to support auditory rehabilitation. In *Proceedings of the XVI International Conference on Human Computer Interaction*, Interaccin '15, pages 10:1–10:8.

[Cheng et al. 2014] Cheng, B. H. C., Eder, K. I., Gogolla, M., Grunske, L., Litoiu, M., Müller, H. A., Pelliccione, P., Perini, A., Qureshi, N. A., Rumpe, B., Schneider, D., Trollmann, F., and Villegas, N. M. (2014). *Using Models at Runtime to Address Assurance for Self-Adaptive Systems*, pages 101–136.

[Dormans 2012] Dormans, J. (2012). The effectiveness and efficiency of model driven game design. In *Entertainment Computing - ICEC 2012*, pages 542–548, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Fernandez et al. 2012] Fernandez, A., Insfran, E., Abrahão, S., Carsí, J. Á., and Montero, E. (2012). Integrating usability evaluation into model-driven video game development. In *Human-Centered Software Engineering*, pages 307–314, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Fluery et al. 2014] Fluery, A., Nakano, D., and Cordeiro, J. H. D. O. (2014). Mapeamento da indústria brasileira e global de jogos digitais. Technical report, BNDES, Brasil.

[Furtado and Santos 2006] Furtado, A. W. B. and Santos, A. L. M. (2006). Using domain-specific modeling towards computer games development industrialization.

[Guana et al. 2015] Guana, V., Stroulia, E., and Nguyen, V. (2015). Building a game engine: A tale of modern model-driven engineering. In *IEEE/ACM 4th International Workshop on Games and Software Engineering*, pages 15–21.

[Guo et al. 2018] Guo, H., Gao, S., Trætteberg, H., Wang, A. I., and Jaccheri, L. (2018). Ontology-based domain analysis for model driven pervasive game development. *Information*, 9(5).

[Guo et al. 2015a] Guo, H., Trætteberg, H., Wang, A. I., and Gao, S. (2015a). A workflow for model driven game development. In *IEEE 19th International Enterprise Distributed Object Computing Conference*, pages 94–103.

[Guo et al. 2015b] Guo, H., Trætteberg, H., Wang, A. I., Gao, S., and Jaccheri, M. L. (2015b). Realcoins: A case study of enhanced model driven development for pervasive games. *Int. Journal of Multimedia and Ubiquitous Engineering*, 5(10):395–410.

[Kasurinen et al. 2017] Kasurinen, J., Palacin-Silva, M., and Vanhala, E. (2017). What concerns game developers? a study on game development processes, sustainability and metrics. In *IEEE/ACM 8th Workshop on Emerging Trends in Software Metrics (WETSoM)*, pages 15–21.

[Kitchenham and Charters 2007] Kitchenham, B. and Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering.

[Matallaoui et al. 2015] Matallaoui, A., Herzig, P., and Zarnekow, R. (2015). Model-driven serious game development integration of the gamification modeling language gaml with unity. In *48th Hawaii Int. Conf. on System Sciences*, pages 643–651.

[Oliveira et al. 2018] Oliveira, A., Bischoff, V., Gonçales, L. J., Farias, K., and Segalotto, M. (2018). Brcode: An interpretive model-driven engineering approach for enterprise applications. *Computers in Industry*, 96:86 – 97.

[Petrillo et al. 2009] Petrillo, F., Pimenta, M., Trindade, F., and Dietrich, C. (2009). What went wrong? a survey of problems in game development. *Comput. Entertain.*, 7(1):13:1–13:22.

[Prado and Lucrédio 2015] Prado, E. F. D. and Lucrédio, D. (2015). A flexible model-driven game development approach. In *2015 IX Brazilian Symposium on Components, Architectures and Reuse Software*, pages 130–139.

[Purushothaman et al. 2017] Purushothaman, S. K., Kashyap, N., Singh, V., Bharti, A., and Sawant, S. (2017). Automation approach for cocos-2dx based multi-player card game for web and mobile. In *2017 2nd International Conference on Computing and Communications Technologies (ICCCT)*, pages 222–225.

[Selic 2014] Selic, B. (2014). Model-based software engineering in industry: Revolution, evolution, or smoke? In *Invited Speaker at Int. Conf. on Industrial Informatics*.

[Sommerville 2010] Sommerville, I. (2010). *Software Engineering (9th Edition)*. Addison-Wesley.

[Suzuki et al. 2011] Suzuki, I., Tsunoda, K., and Hishiyama, R. (2011). Game description language and frameworks for langrid gaming. In *Proceedings of the 29th ACM International Conference on Design of Communication*, SIGDOC '11, pages 67–74.

[Sánchez et al. 2015] Sánchez, K., Garcés, K., and Casallas, R. (2015). A dsl for rapid prototyping of cross-platform tower defense games. In *2015 10th Computing Colombian Conference (10CCC)*, pages 93–99.

[Whittle et al. 2015] Whittle, J., Hutchinson, J., Rouncefield, M., Håkan, B., and Rogardt, H. (2015). A taxonomy of tool-related issues affecting the adoption of model-driven engineering. *Software & Systems Modeling*, pages 1–19.

[Wohlin et al. 2012] Wohlin, C., Runeson, P., Höst, M., Ohlsson, M., Regnell, B., and Wesslén, A. (2012). *Experimentation in Software Engineering*. Springer.

[Zhu 2014] Zhu, M. (2014). *Model-Driven Game Development Addressing Architectural Diversity and Game Engine-Integration. Disponível em <https://pdfs.semanticscholar.org/195e/55c90fd109642116ee51f7205c106f341111.pdf>*. PhD thesis.

[Zhu et al. 2016] Zhu, M., Wang, A. I., and Trætteberg, H. (2016). Engine-cooperative game modeling (ecgm): Bridge model-driven game development and game engine tool-chains. In *Proceedings of the 13th International Conference on Advances in Computer Entertainment Technology*, ACE '16, pages 22:1–22:10.