**استفاده از رویکرد مدل‌رانده در بازی‌وارسازی دوره‌های آموزشی**

**سیده هستی موسوی دهویی1، شکوفه کلاه‌دوز رحیمی\*2 ، لیلا صمیمی دهکردی۳**

**۱-دانشجوی کارشناسی ارشد مهندسی کامپیوتر-گروه مهندسی نرم‌افزار، دانشگاه اصفهان**

Hasti.mousavi97@eng.ui.ac.ir

**۲-استادیار و عضو هیئت‌علمی دانشکده مهندسی کامپیوتر- گروه مهندسی نرم‌افزار، دانشگاه اصفهان**

Sh.rahimi@eng.ui.ac.ir

**۳- مدرس دانشكده فني و مهندسي- گروه مهندسي كامپيوتر، دانشگاه شهركرد**

samimi@eng.ui.ac.ir

**خدایا به خاطر همه عواملی که باعث میشه مقالم قبول نشه در کنفراس بازی ها ی اصفهان متاسفم لطفا منو ببخش متشکرم دوستت دارم . هدایا میدونم که همه ی مسءولیتش متوجه خودمه متاسفم لطفا منو ببخش متشکرم دوستت دارم**

چکیده

اهمیت آموزش در دنیای امروز بر هیچ‌کس پوشیده نیست. امروزه مراکز آموزشی به کمک پیشرفت فناوری، درصدد آموزش فراگیران خود به شیوه‌های الکترونیکی هستند. در این مسیر، متولیان آموزشی با مشکلاتی ازجمله، زیاد بودن مواد آموزشی، مدت‌زمان یادگیری محدود، عدم حضور رسانه‌های تعاملی و پایین بودن کیفیت آموزش مواجه هستند که میزان انگیزه و تعهد دانشجویان را تحت تاثیر منفی قرار داده است. راه‌حلی که در این زمینه می‌تواند بسیار مؤثر باشد استفاده از بازی‌وارسازی[[1]](#footnote-1) در دامنه‌ی آموزش الکترونیکی است. از طرفی توسعه‌دهندگان نرم‌افزارها، برای تولید دوره‌هایی که بتواند از عناصر بازی برای آموزش استفاده کنند، با پیچیدگی‌های زیادی ازجمله، عدم آشنایی با مفاهیم دقیق این دامنه مواجه هستند که این عامل روند توسعه‌ی این نرم‌افزارها را زمان‌بر می‌کند. در این مقاله تلاش ما بر این است که به کمک رویکرد مدل‌رانده، با مدل‌سازی مفاهیم بازی‌وارسازی، فرایند طراحی و پیاده‌سازی دوره‌ها را برای توسعه‌دهندگان آسان‌ترکنیم. در این راستا فرامدلی برای این دامنه با استفاده از چارچوب‌ ایکلیپس[[2]](#footnote-2) طراحی کرده‌ایم و سپس مدلی مبتنی بر این فرامدل تولید شده است که نشان- دهنده‌ي یک دوره‌ی بازی‌وارسازی می‌باشد.

**کلمات کليدي: بازی‌وارسازی، مدل‌رانده، آموزش**

**1-مقدمه**

در دنیای امروز، آموزش یکی از مهم‌ترین نیازمندی‌های جامعه‌ی انسانی محسوب می‌شود. تاریخ نشان داده است که پیشرفت بشر، حاصل یادگیری و دانش یا آموزش است. آموزش موضوعاتی که حجم و مواد آموزشی بالایی دارند، همواره موضوعی چالش‌برانگیز برای فراگیران و اساتید بوده است [۱].

از سوی دیگر، نادیده انگاشتن تفاوت‌های فردی و مدت‌زمان محدود آموزش، انگیزه دانشجویان را کاهش داده است. پس متولیان آموزشی با دو مشکل اساسی در این بخش نظیر کاهش انگیزه دانشجویان و زمان کم و حجم بالای مواد درسی مواجه هستند. در اینجا استفاده از بازی‌وارسازی و الکترونیکی کردن دوره‌های آموزشی می‌تواند با تأثیر مثبت خود بر یادگیری، انگیزه را افزایش، مدت‌زمان لازم برای فرایند تدریس را کاهش و یادگیری حجم بالای مواد آموزشی را برای فراگیران امکان‌پذیرتر ‌کند [۱].

در این راستا، نیاز به طراحی و پیاده‌سازی دوره‌های آموزشی که بتواند از بهترین عناصر بازی برای بازی‌وارسازی این دوره‌ها بهره گیرد، احساس می‌شود. این امر نیازمند تخصص در حوزه بازی‌ها و تولید دوره‌های آموزشی است. ازاین‌رو تلاش ما بر این است که به کمک رویکرد مدل‌رانده، با مدل‌سازی بازی‌وارسازی در تولید دوره‌های الکترونیکی آموزشی، سطح انتزاع را بالابرده و توسعه‌دهندگان نرم‌افزاری را از فهم جزییاتی که با آن آشنا نیست بی‌نیاز کنیم؛ بنابراین عناصر کلیدی که در تولید دوره‌های آموزشی بازی‌‌سازی شده به‌کار می‌روند، استخراج ‌شده و فرامدلی بر اساس این مفاهیم طراحی شده است. سپس به عنوان نمونه،‌ مدلی که از این فرامدل تبعیت می‌کند، تولید شده است. به کمک فرامدل، مدل‌های گوناگونی از این دوره‌ها توسط توسعه‌دهندگان نرم‌افزاری طراحی می‌شود. در مراحل بعدی این مدل‌ها می‌توانند توسط برنامه‌های تبدیل مدل به کد، به کد لازم برای پیاده‌سازی در سکوهای مقصد تبدیل شوند.

**۲-مروری بر ادبیات تحقیق**

قلمرو پژوهش حاضر از سه دامنه‌ی مهندسی نرم‌افزار مدل‌رانده[[3]](#footnote-3)، آموزش و بازی‌وارسازی تشکیل شده است که در ادامه برای فهم بهتر جزيیات بیان شده، به معرفی ادبیات موردنیاز این مقاله پرداخته می‌شود.

**۱-۲-** **مهندسی نرم‌افزار مدل‌رانده**

توسعه نرم‌افزار شامل فعالیت‌هایی است که برای افزایش کیفیت نرم‌افزار و بالا بردن سرعت تولید نرم‌افزار برنامه‌ریزی می‌شود. این فرآیند همواره برای مهندسان نرم‌افزار دارای پیچیدگی‌هایی بوده است. با کمک متدولوژی‌هایی که برای توسعه نرم‌افزار تعریف شده تا حدود بسیار زیادی از پیچیدگی و سختی فرایند توسعه نرم‌افزار کاسته شده است [۲]. یکی از این متدولوژی‌ها که در سال‌های اخیر با اقبال زیادی از سوی مهندسان نرم‌افزار مواجه شده است، مهندسی نرم‌افزار مدل‌رانده است. MDSE را می‌توان به‌عنوان یک متدولوژی برای استفاده از مزایای مدل‌سازی در فعالیت‌های مهندسی نرم‌افزار تعریف کرد. مطالعات مختلف کمی و کیفی، نشان داده است که روش‌های MDSE باعث افزایش کارایی و اثربخشی در توسعه نرم‌افزار شدند [۲].

در مهندسی مدل‌رانده، یک فرمول ساده نرم‌افزار را ایجاد می‌کند [۲].

مدل‌ها + تبدیل‌ها[[4]](#footnote-4) = نرم‌افزار

بدیهی است که هر دو، مدل‌ها و تبدیل‌ها باید در قالب برخی نمادها[[5]](#footnote-5) بیان شوند که در MDSE، یک زبان مدل‌سازی[[6]](#footnote-6) نامیده می‌شود [۲].

مهمترین مفاهیم مدل‌رانده عبارتند از:

* مدل: مدل را می‌توان نمایشی از یک سیستم دانست که با حذف جزییات غیرضروری باعث می‌شود تا درک کاربر از سیستم افزایش یابد [۲].
* فرامدل[[7]](#footnote-7)‌: مدل‌هایی هستند که عناصر مدل‌سازی را توصیف می‌کنند. به‌طور دقیق‌تر، یک فرامدل ساختار احتمالی مدل‌ها را به روشی انتزاعی توصیف می‌کند و ساختارهای یک‌زبان مدل‌سازی و روابط آن‌ها را تعریف می‌کند [۳].
* تبدیل: برنامه‌هایی هستند که مدل‌ها را مطابق قوانین خاصی که در آن تعریف می‌شود، به مدل‌های سطح پایین‌تر یا بالاتر تبدیل می‌کنند [۲].
* زبان مدل‌سازی خاص دامنه: زبان مدل‌سازی خاص دامنه برای شناسایی مفاهیم یک دامنه خاص به کار می‌رود. این کار باعث می‌شود که بر روی یک دامنه تمرکز بیشتری داشته و بتوان تمام ویژگی و قابلیت‌های آن دامنه را استخراج کرد [۴].

هدف توسعه‌دهندگان نرم‌افزار، تولید کدی باکیفیت بالاست که در این رویکرد می‌توان بسیاری از مراحل و محصولات توسعه نرم‌افزاری را به‌صورت (نیمه) خودکار تولید نمود که این کار منجر به بالا رفتن کیفیت کد و محصول نرم‌افزاری می‌شود. فرایند توسعه در مهندسی نرم‌افزار مدل‌رانده در شکل ۱ نشان داده شده است. ابتدا متخصص مدل‌رانده مفاهیم اصلی دامنه‌ها را استخراج کرده و با استفاده از نرم‌افزار ایکلیپس،‌ فرامدل از عناصر اصلی دامنه‌ها و روابط بین آن‌ها طراحی می‌شود. سپس با استفاده از یک چارچوب ایجاد زبان مدل‌سازی مانند ایکستکست[[8]](#footnote-8)، زبان مدل‌سازی خاص دامنه ساخته می‌شود. پس‌ازآن با استفاده از نرم‌افزارهایی مانند سیریوس[[9]](#footnote-9)، ویرایشگر گرافیکی برای تولید گرافیکی مدل‌ها ایجاد می‌شود. در مرحله بعد با استفاده از یک زبان تبدیل مناسب مانند ای‌تی‌ال[[10]](#footnote-10) یا اکسلئو[[11]](#footnote-11)، برنامه‌هایی برای تبدیل مدل‌های ایجاد شده به کد قابل پیاده‌سازی در سکوهای مقصد مانند اندروید، سکوهای تحت وب و غیره تولید می‌شود. این کد می‌تواند به‌صورت (نیمه) خودکار تولید شود.

MetaModel

طراحی زبان خاص دامنه

طراحی نحو گرافیکی زبان

تبدیل مدل به مدل یا مدل به کد

**شکل ۱- فرایند توسعه نرم‌افزار در مهندسی مدل‌رانده**

**۲-۲- بازی‌وارسازی**

بازی‌ها از دیرباز به‌عنوان یکی از فعالیت‌های لذت‌بخش و خوشایند بشر شناخته‌ شده است. بازی‌ها، فعالیت‌هایی هدف محور هستند که مطابق با قوانینی که برای بازی تعریف می‌شود، ‌صورت می‌پذیرند [۵, ۶]. بازی‌وارسازی را می‌توان استفاده از عناصر طراحی بازی در زمینه‌های غیر بازی تعریف کرد [۶].

در طراحی بازی‌وارسازی طبقه‌بندی‌های مختلفی وجود دارد و یکی از معروف‌ترین آن‌ها چارچوب مکانیک–دینامیک-زیبایی‌شناسی[[12]](#footnote-12) است. در این چارچوب توجه به ابعاد مکانیک، دینامیک و زیبایی‌شناسی[[13]](#footnote-13) از اهمیت بالایی برخوردار است [۶].

* مکانیک بازی: اجزای خاص بازی را در سطح نمایش داده‌ها و الگوریتم‌ها توصیف می‌کند. مکانیک بازی ممکن است به‌شدت بر انگیزه و تعهد کاربر تأثیر بگذارد. در اینجا ذکر این نکته ضروری است که مکانیک بازی با قوانین بازی متفاوت است. قوانین بازی‌ رفتارهای تأیید شده‌ای را تعیین می‌کند که در هنگام اجرای مکانیک مربوط به آن، دنبال می‌شوند. متداول‌ترین عناصر مکانیک بازی عبارت‌اند از: امتیاز[[14]](#footnote-14)، جدول امتیاز[[15]](#footnote-15)، مراحل[[16]](#footnote-16)، سیستم دستاورد[[17]](#footnote-17).
* دینامیک بازی: رفتارهای زمان اجرای کاربران بر روی مکانیک‌های بازی را توصیف می‌کنند؛ بنابراین دینامیک دلیل رفتار انگیزشی کاربر نسبت به مکانیک بازی را نشان می‌دهد. رایج‌ترین دینامیک بازی شامل عناصر پاداش‌ها[[18]](#footnote-18)، موقعیت[[19]](#footnote-19)، دستاورد[[20]](#footnote-20)، خود ابرازی[[21]](#footnote-21)، رقابت[[22]](#footnote-22)، نوع‌دوستی[[23]](#footnote-23) می‌باشد.
* زیبایی‌شناسی: پاسخ‌های احساسی مطلوب برانگیخته ‌شده در بازیکن را در هنگام تعامل با سیستم بازی، توصیف می‌کند. زیبایی‌شناسی شامل عناصری مانند احساس، روایت، چالش، همکاری دوستانه، اکتشاف، بیان و تفویض[[24]](#footnote-24) است.

یکی از کاربردهای بازی‌وارسازی، استفاده از آن برای تغییر رفتار یک فرد یا افراد یک جامعه است [۷]. درواقع بازی‌وارسازی را می‌توان فناوری دانست که برای تأثیرگذاری بر رفتار کاربر بدون مجبور کردن فرد به تغییر رفتار ایجاد شده است. برای تأثیرگذاری بر رفتار، باید درک کنیم که رفتار چگونه ایجاد می‌شود و چه چیزی بر رفتار تأثیر می‌گذارد. عوامل موثر در ایجاد و تغییر رفتار عبارتند از:

* انگیزه[[25]](#footnote-25): انگیزه عامل مهمی است که باید در بازی‌سازی مورد توجه قرار گیرد، به‌خصوص به این دلیل که باعث پیش بردن رفتار انسان می‌شود. انگیزه میل به انجام کاری است و می‌توان آن را به دو نوع تقسیم‌بندی کرد: درونی[[26]](#footnote-26) و بیرونی[[27]](#footnote-27) [۷].
* تعهد[[28]](#footnote-28): نشان‌دهنده اشتیاق و درگیری عاطفی در شرکت و تکمیل فعالیت‌ها است [۸]. اینکه یک فرد چقدر می‌تواند خود را درگیر انجام کاری قرار دهد، میزان تعهد شخص را مشخص می‌کند.

محققان در مورداستفاده از عناصر بازی‌سازی دریافته‌اند که به‌کارگیری این عناصر تأثیر مثبتی بر تعهد، انگیزه و عملکرد کلی فراگیران از طریق بازخورد فوری[[29]](#footnote-29) و همکاری[[30]](#footnote-30) دارند [۸].

در بخش بعدی مقاله، تاریخچه‌ای از پژوهش‌هایی که در این زمینه انجام شده است،‌ مورد بررسی قرار می‌گیرد.

**۳-پیشینه تحقیق**

در ادامه کارهای تحقیقاتی انجام شده در زمینه‌ي آموزش و بازی‌وارسازی و کاربرد رویکرد مدل‌رانده در این دامنه‌ها بررسی و تحلیل می‌شوند.

کازنتینو[[31]](#footnote-31) و همکاران [۹]، در سال ۲۰۱۷،‌ یک رویکرد مدل‌رانده برای پشتیبانی از بازی‌وارسازی آموزش ارائه‌ داده‌اند. در این پژوهش برای مدل‌سازی محیط بازی‌وارسازی یک فرامدل بازی و دو فرامدل مکمل، یکی مربوط به مصنوعات نرم‌افزار و دیگری مربوط به وضعیت بازیکن در بازی و همچنین، یک نحو متنی برای تعریف اجزا و مکانیک بازی ارائه ‌شده است. چارچوب ارائه ‌شده در این کار، داده‌های پروژه را هنگامی‌که عملی اتفاق می‌افتد، جمع‌آوری کرده، مکانیک بازی را اجرا می‌کند. سپس داده‌های تولید شده بازی‌وارسازی را پردازش می‌کند و در انتها اطلاعات مربوط به بازی را نمایش می‌دهد.

در این رویکرد به کمک رمزگذاری بر روی مدل‌های وضعیت به مشکلات ناشی از تقلب و حریم شخصی توجه نشان داده‌ شده است. چارچوب معرفی شده در این پژوهش، در زمینه‌ی بازی‌وارسازی آموزش UML اعمال‌ شده است. هرچند برای یادگیری زبان‌هایی مانند SQL نیز به کار می‌رود. یکی از نقطه ‌ضعف‌های این کار را می‌توان عدم توجه به سایر جنبه‌های بازی‌وارسازی مانند رقابت و میزان پیشرفت دانست که این کار به کمک عناصری مانند تابلو امتیازات و نوار پیشرفت می‌توانست اعمال شود. همچنین این چارچوب خاص آموزش UML طراحی شده است و کاربری محدودی دارد.

دوپرادو[[32]](#footnote-32) و لوکردیو[[33]](#footnote-33) [۱۰]، یک رویکرد توسعه مدل‌رانده بازی[[34]](#footnote-34) را برای تولید بازی‌های الکترونیکی ارائه‌ کرده‌اند که در این روش چندین زبان خاص دامنه را با الگوهای طراحی و موتور بازی ترکیب می‌کند تا کد تولید شده بتواند با کد دستی ادغام شود. این کار سه زبان خاص دامنه و سه ویرایشگر برای سه جنبه مختلف دوربین، شخصیت و سناریو ارائه می‌دهد. توسعه‌دهنده می‌تواند مدل خود را با توجه به این جنبه‌ها تولید ‌کند. سپس مدل‌ها در قالب فایل XML ذخیره ‌شده و توسط پردازشگر قالب Freemaker خوانده ‌شده و مطابق قالب‌ها،‌ کد پیاده‌سازی را تولید می‌کند. کد جاوا تولیدشده به کمک بعضی از الگوهای طراحی با کد دستی ادغام شده و بر روی موتور Jmonkey اجرا می‌شود. به طور کلی مجموعه ابزارها و مولد کدها با نام jMEGenerator ارائه ‌شده است که یک ابزار آموزشی برای مدیریت دوره درسی و ارتباط دادن فراگیران و منابع آموزشی با فعالیت‌های یادگیری است.

در پژوهشی دیگر، جورگلیتیس[[35]](#footnote-35) و همکاران [۱]، برای آموزش یوام‌ال یک فرامدل ارائه داده‌اند. این فرامدل ساختار درس و عناصر اصلی بازی‌وارسازی را مشخص می‌کند. در فرامدل پیشنهادی بخش معنا برای آموزش اصول متدولوژی فرایند یکپارچه[[36]](#footnote-36) تعریف ‌شده است و بخش نحو برای آموزش نمودارهای مختلف UML استفاده ‌شده است. برای بازی‌وارسازی آموزش UML یک ساختار سطح‌بندی شده ارائه ‌شده است که در هر سطح نمودارهای خاص و بخشی از متدولوژی فرایند یکپارچه آموزش داده می‌شود و تا زمانی که سطح قبل پایان نیافته و امتیاز لازم کسب نشده است، اجازه دسترسی به محتوای سطح بالاتر وجود ندارد. عناصر بازی استفاده ‌شده در این درس شامل امتیاز و انواع پاداش مانند: سکه، نشان، تخته امتیازات، قفل‌کردن محتوا و تجارت است. درس بازی‌وارسازی شده بر روی سکوی مودل[[37]](#footnote-37) پیاده‌سازی شده است.

در ارزیابی این کار نمرات نیم‌سال بهار در دو سال متوالی بررسی ‌شده است اما مشخص نشده است که این نمرات بر چه اساسی تعیین ‌شده و امتیازات و نشان‌ها و سایر دستاوردها چه نقشی در محاسبه این نمره داشته‌اند.

ساویچ[[38]](#footnote-38) و همکاران [۱۱]، سعی کردند اصول مدل‌رانده را برای توسعه و مدیریت دوره الکترونیکی اعمال کنند. نویسندگان به کمک مدل مستقل از سکو، امکان ذخیره و مدیریت دوره‌ها را فراهم می‌کنند. مدل مبدأ شامل یک سری مؤلفه‌ی مجزا است که به روشی قابل‌خواندن برای ماشین[[39]](#footnote-39) تعریف ‌شده ‌که هر مؤلفه جنبه‌ی خاصی را نشان می‌دهد. مدل مبدأ ازمؤلفه‌ی لرنینگ آبجکتیو[[40]](#footnote-40)، لرنینگ سورس[[41]](#footnote-41) و استراتژی اینستراکشن[[42]](#footnote-42) تشکیل ‌شده است. مدل از فرازبان طراحی آموزشی یادگیری الکترونیکی[[43]](#footnote-43) استفاده می‌کند. این زبان یک فرا زبان مبتنی بر xml است که یک نحو انتزاعی را برای تعریف قوانین انتخاب و سازمان‌دهی دوره‌ها فراهم می‌کند [30]. این مدل مستقل از سکو به ۴ مدل خاص سکو[[44]](#footnote-44) به فرمت‌های IMS learning design ,[[45]](#footnote-45) SCORM ,[[46]](#footnote-46)LAMS و[[47]](#footnote-47)Sakai می‌تواند تبدیل می‌شود. همچنین، نویسندگان یک معماری برای سیستم مدیریت دوره ارائه داده اند. نویسندگان یک مطالعه موردی انجام داده‌اند که در آن درس برنامه‌نویسی وب مدل‌سازی شده و سپس به چهار مدل خاص سکو ذکر شده، تبدیل‌ شده است.

در تحقیقی دیگر، آراجو[[48]](#footnote-48) و همکاران [۱۳]، یک بازی دیجیتال آموزشی به نام ریداکت[[49]](#footnote-49) ارائه داده‌اند که یک رویکرد جدیدی برای آموزش مفاهیم پایه برنامه‌نویسی مبتدی جاوا اسکریپت نیز در بردارد. این مفاهیم شامل توابع، ‌نوع داده بولین،‌ عملگرهای مقایسه‌ای، عبارت‌های شرطی و نگاشت توابع بر روی مجموعه‌ها بود. در این کار از بلوک گرافیکی برای طراحی مسائل استفاده‌ شده و از دانشجو خواسته می‌شود تا با توجه به جواب مورد انتظار، در جای خالی کد مناسب را بنویسد. ‌بازیکن کد نوشته ‌شده توسط خودش را اجرا می‌کند و به همین ترتیب ۷۲ سطح طراحی‌ شده در این بازی را تا هرزمانی که بخواهد، انجام می‌دهد. پژوهشگران این مقاله دو ارزیابی آزمایشگاهی و آنلاین برای این بازی‌ ترتیب دادند که از نظر کیفیت و جذابیت یادگیری با استقبال زیادی از سوی شرکت‌کنندگان مواجه شد.

در مطالعه‌ای به نام چارچوب طراحی بازی‌وارسازی توسط مهندسی مدل محور، بوشیارون[[50]](#footnote-50) [۱۴]، یک چالش مهم در توسعه و تکامل سیستم‌های بازی‌وارسازی را عدم امکان بازبینی یا معرفی عناصر و مکانیک‌های جدید بازی در حین اجرای سیستم می‌داند؛ بنابراین یک چارچوب طراحی بازی‌وارسازی که متشکل از زبان‌های خوش‌تعریفی برای طراحی بازی و اجزای آن و جزئیات رفتاری است، ‌ارائه شده است. این چارچوب دارای یک معماری لایه‌ای است که بالاترین لایه، پایه و اساس سایر لایه‌ها را تشکیل می‌دهد. این معماری شامل ۵ زبان GML[[51]](#footnote-51) که عناصر اساسی واصلی مربوط به سیستم‌های بازی‌وارسازی را معرفی می‌کند، زبان GaML[[52]](#footnote-52) که جزییات بیشتری را درباره چگونگی مونتاژ کردن اجزای بازی برای ایجاد یک برنامه کاربردی بازی‌وارسازی، اعمال می‌کند، GiML[[53]](#footnote-53) برای ساختن نمونه‌های مختلفی از یک بازی که در GaML تعریف ‌شده است، استفاده می‌شود. GsML[[54]](#footnote-54) برای شبیه‌سازی تغییر حالت‌هایی است که در بازی رخ می‌دهد و GadML[[55]](#footnote-55)، شامل یک سیستم پیشنهاددهنده است که می‌تواند با توجه به سطح مهارت و تاریخچه بازیکن، چالش ایجاد کند.

در کارهای بررسی شده در بخش پیشینه، از عناصر بازی محدودی استفاده شده است که کار ما از تنوع بالاتری برخوردار است. همچنین‌کارها به طور خاص برای آموزش یا چارچوب بازی‌وارسازی استفاده شده است. در حالی که در کار ما می‌توان با ترکیب آموزش،‌ بازی‌وارسازی و مدل‌رانده، آموزش در زمینه‌های مختلف را فراهم کرد.

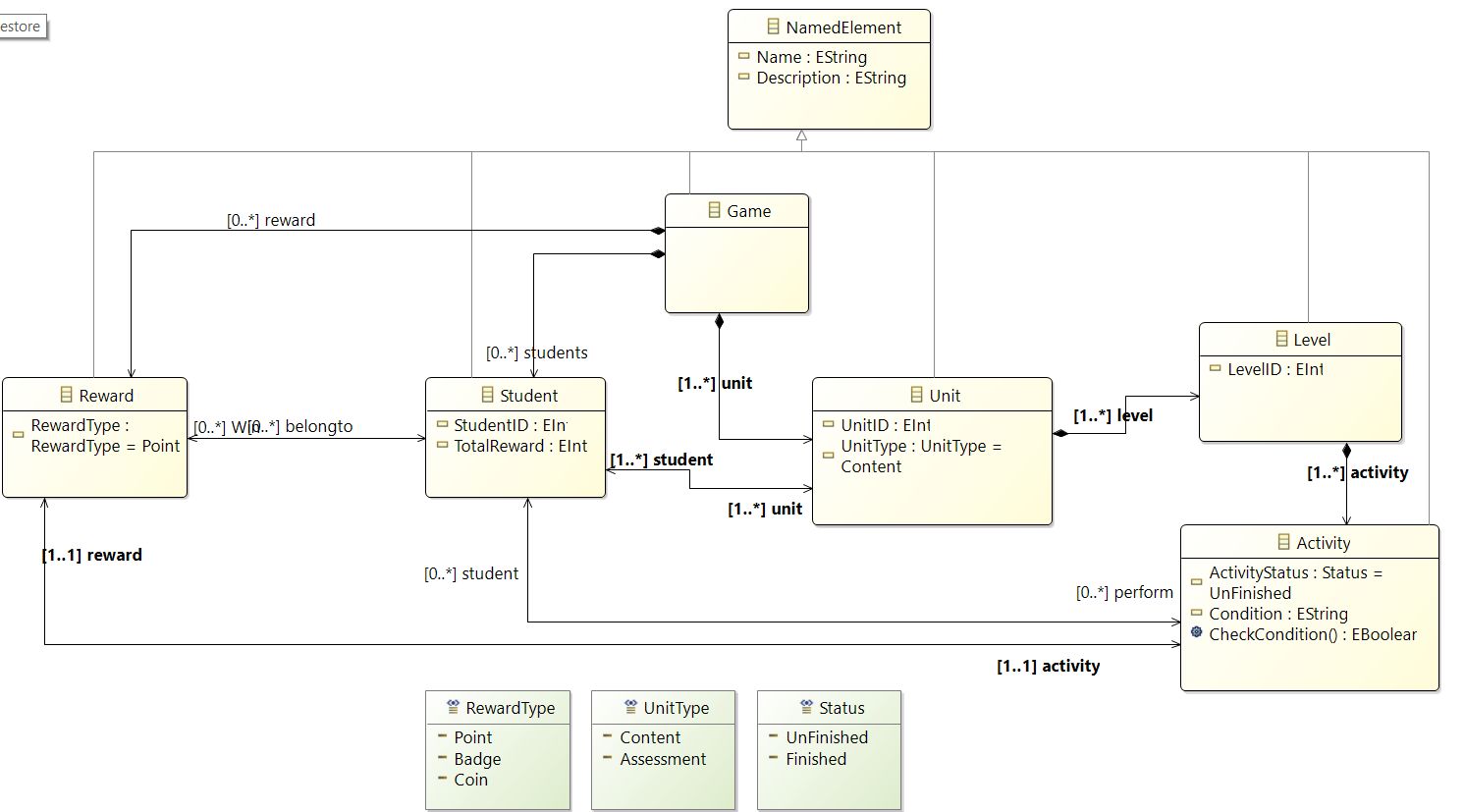
**۴-روش تحقیق**

در این مقاله فرامدلی برای دامنه بازی‌وارسازی در زمینه‌ی آموزشی طراحی شده است که شامل مهم‌ترین مفاهیم لازم برای مدل‌سازی این دامنه می‌باشد. در ادامه به معرفی فرامدل و عناصر آن پرداخته می‌شود.

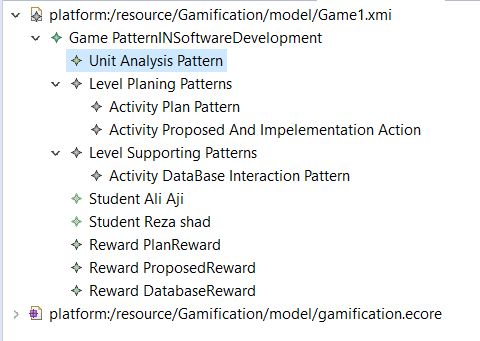
**۴-۱- فرامدل**

پس از بررسی کردن دامنه‌ی بازی‌وارسازی و دامنه‌ی آموزش الکترونیکی، مفاهیم زیر به شکل عناصر فرامدل و ویژگی‌های آن ها استخراج شده است که همه‌ی این موجودیت‌ها و ارتباط بین آن‌ها در شکل ۲ نشان داده شده است.

* NamedElement: این عنصر دارای ویژگی نام و توضیحات[[56]](#footnote-56) است و تمام عناصر موجود در فرامدل، از این کلاس ارث‌بری می‌کنند.
* Game: این عنصر، کلاس ریشه محسوب می‌شود که از این کلاس به‌تمامی عناصر فرامدل می‌توان دسترسی داشت.
* Unit: هر کلاس game می‌تواند شامل ۱ یا چند Unit (بخش) باشد. این کلاس دارای ویژگی UnitStatus است که از نوع داده شمارشی به اسم Status است و تکمیل یا عدم تکمیل مرحله را مشخص می‌کند. همچنین ویژگی UnitID که شناسه‌ی منحصربه‌فرد هر Unit را نشان می‌دهد. ویژگی UnitType نیز نوع بخش طراحی ‌شده را مشخص می‌کند که می‌تواند به‌ صورت محتوای درسی یا ارزشیابی از محتوا باشد.
* Level: هر کلاس Unit می‌تواند شامل ۱ یا چند Level (مرحله) باشد که مراحلی را که یک کاربر باید طی کند، تعریف می‌کند. این عنصر دارای ویژگی LevelStatus است که کارکردی مشابه ویژگی UnitStatus در عنصر Unit دارد.
* Activity: این عنصر فعالیت‌هایی که باید در یک سطح انجام شود را مشخص می‌کند. هر Levelمی‌تواند شامل یک یا چند فعالیت باشد. این عنصر نیز دارای ویژگی ActivityStatus، ActivityID و Condition است که به ترتیب، وضعیت فعالیت، شناسه فعالیت و شرط لازم برای تکمیل فعالیت را مشخص می‌کنند. این کلاس با کلاس Reward در ارتباط است زیرا به ازای هر فعالیتی که توسط کاربر تکمیل می‌شود،‌ پاداشی به او تعلق می‌گیرد. متد CheckCondition نیز مشخص می‌کند شرط برآورده شده است یا خیر.
* Reward: این عنصر نشان‌دهنده پاداشی است که کاربر دریافت می‌کند. ویژگی RewardType این کلاس نوع پاداش اعطایی را مشخص می‌کند و از نوع داده شمارشی به همین نام است که می‌توان امتیاز[[57]](#footnote-57)، نشان[[58]](#footnote-58) و سکه[[59]](#footnote-59) باشد.
* Student: این عنصر کاربر این برنامه را نشان می‌دهد که دارای شناسه مخصوص هر کاربر و جمع پاداش‌هایی است که تاکنون دریافت کرده است. این عنصر با عنصر Activity در یک ارتباط چند به چند قرار دارد، ‌زیرا کاربر می‌تواند فعالیت‌های مختلفی را انجام دهد و همچنین فعالیت می‌تواند توسط چند کاربر انجام شود.

**** همچنین این عنصر با عنصر Reward نیز در ارتباط است که مشخص می‌کند، هر کاربر چه پاداش‌هایی دریافت کرده است. به‌علاوه هر کاربری باید مشخص شود که در چه بخشی مشغول به فعالیت است و کدام بخش را تکمیل کرده است.

**شکل ۲- فرامدل بازی‌وارسازی آموزش**

با استفاده از این فرامدل‌های مختلفی را می‌توان تولید کرد. فرمت XMI یکی از مدل‌های ساخته ‌شده بر اساس این فرامدل در شکل ۳ نشان داده ‌شده است. این مدل بازی‌وارسازی برای آموزش الگوها در مهندسی نرم‌افزار را نشان می‌دهد؛ که از یک بخش به نام Analysis Pattern تشکیل شده که این بخش شامل دو مرحله Planning و Supporting Pattern است. برای مرحله‌ی اول دو فعالیت و برای مرحله‌ي دوم یک فعالیت متناسب با آن‌ها طراحی شده است. همچنین دو کاربر علی و رضا در این بخش درسی شرکت نموده و مراحل و فعالیت‌های موردنظر را باید به اتمام برسانند. برای هر فعالیت انجام شده نیز پاداش متناظر با آن تعریف شده است که در صورت برقراری شرط تعیین شده به کاربر داده می‌شود.

**شکل ۳- مدل XMI تولید شده از فرامدل**

هدف نهایی در مهندسی نرم‌افزار مدل‌رانده تولید کد با کیفیت به‌صورت خودکار یا نیمه‌خودکار است؛ بنابراین از مدل ساخته شده در شکل ۳ میتوان برای تولید مدل‌های با سطح انتزاع پایین‌تر یا کد استفاده کرد. ازآنجایی‌که مدل‌سازی به‌صورت متنی برای کاربران امری خسته‌کننده است و امکان اشتباه در مدل‌سازی را بالاتر می‌برد، با ایجاد ویرایشگر گرافیکی مانند شکل ۴، مدل‌سازی برای کاربر راحت‌تر خواهد بود. پس از مدل‌سازی، با استفاده از زبان‌های تبدیل، برنامه‌هایی برای تبدیل مدل به مدل یا مدل به کد نوشته می‌شوند. کدهای بازی‌وارسازی ایجاد شده در هر سکوی قابل پشتیبانی می‌توانند پیاده‌سازی شوند.

Palette

Game

Unit

Level

Reward

Student

Activity

Pattern in Software Engineering

Analysis Pattern

Supporting

Database

Interaction

Plan

Proposed Action

Plan

Proposed

Database

Ali

Reza

Planning

**شکل ۴- نمونه‌ای از ویرایشگر گرافیکی**

**۴-نتیجه‌گیری و کارهای آینده**

آموزش در دنیای کنونی جایگاه بسیار ویژه‌ای دارد و همواره متولیان آن درصدد بهبود کیفیت آموزش با به‌کارگیری تکنولوژی‌های نوین هستند. ازاین‌رو بسیاری از مراکز آموزشی به دنبال ارائه دوره‌های آموزشی خود به‌صورت الکترونیکی هستند. از سوی دیگر آموزش با چالش‌هایی مانند بی‌انگیزگی فراگیران دست‌وپنجه نرم می‌کند. در این پژوهش سعی شده با به‌کارگیری بازی‌وارسازی، ‌فرایند یادگیری جذاب‌تر شده و بتواند به کمک رویکرد مدل‌رانده فرایند توسعه دوره‌های الکترونیکی را که به‌طورمعمول در سکوهای تحت وب پیاده‌سازی می‌شوند، آسان‌ترکند. ازاین‌رو فرامدلی جهت بازی‌وارسازی آموزش ارائه‌ شده است. مدل‌های مختلفی از این فرامدل، با توجه به نیازهای مشتری تولید شده که این مدل‌ها می‌توانند مورداستفاده برنامه‌های تبدیلی که به‌منظور تولید کد پیاده‌سازی دوره‌های الکترونیکی نوشته ‌شده است، قرار گیرد.

برای کارهای آتی، می‌توان ویرایشگری را برای تولید دوره‌ها به‌صورت گرافیکی، ایجاد کرد تا مدل‌سازی را برای کاربر آسان‌تر کند. همچنین می‌توان از عناصر بازی‌ معرفی شده در این پژوهش در دامنه‌ی سلامت در بخش‌های مختلف و برای رده‌های سنی متفاوت استفاده کرد. به‌ عنوان‌مثال، میتوان افزایش انگیزه در زمینه‌ی تناسب‌اندام و یا مصرف مناسبی از مواد غذایی را در این زمینه نام برد. به‌علاوه برای تولید دوره‌های آموزشی بازی‌سازی شده در حوزه‌ی پزشکی نیز می‌توان از رویکرد به‌کاررفته در این پژوهش استفاده کرد.

**۵-منابع**

[1] M. Jurgelaitis, L. Čeponienė, J. Čeponis, and V. Drungilas, "Implementing gamification in a university‐level UML modeling course: A case study," *Computer Applications in Engineering Education*, vol. 27, no. 2, pp. 332-343, 2019.

[2] M. Brambilla, J. Cabot, and M. Wimmer, "Model-driven software engineering in practice," *Synthesis Lectures on Software Engineering*, vol. 1, no. 1, pp. 1-182, 2012.

[3] M. Völter, T. Stahl, J. Bettin, A. Haase, and S. Helsen, Model-driven software development: technology, engineering, management. John Wiley & Sons, 2013.

[4] U. Frank, "Domain-specific modeling languages: requirements analysis and design guidelines," in *Domain Engineering*: Springer, 2013, pp. 133-157.

[5] F. Groh, "Gamification: State of the art definition and utilization," Institute of Media Informatics Ulm University, pp. 39-46, 2012.

[6] A. Matallaoui, N. Hanner, and R. Zarnekow, "Introduction to gamification: Foundation and underlying theories," in *Gamification*: Springer, 2017, pp. 3-18.

[7] A. AlMarshedi, V. Wanick, G. B. Wills, and A. Ranchhod, "Gamification and behaviour," in *Gamification*: Springer, 2017, pp. 19-29.

[8] M. J. Molumby, "Effects of gamification on motivation and engagement in secondary curriculum," University of Northern Iowa, pp.1-34, 2016.

[9] V. Cosentino, S. Gérard, and J. Cabot Sagrera, "A model-based approach to gamify the learning of modeling," *CEUR Workshop Proceedings*, 2017.

[10] E. F. do Prado and D. Lucrédio, "A flexible model-driven game development approach," in IX *Brazilian Symposium on Components, Architectures and Reuse Software*, 2015, pp. 130-139.

[11] G. Savić, M. Segedinac, D. Milenković, T. Hrin, and M. Segedinac, "A model-driven approach to e-course management," *Australasian Journal of Educational Technology*, vol. 34, no. 1, pp. 14-29, 2018.

[12] I. Arawjo, C.-Y. Wang, A. C. Myers, E. Andersen, and F. Guimbretière, “Teaching Programming with Gamified Semantics,” *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems - CHI 17*, 2017.

[13] A. Bucchiarone, A. Cicchetti, and A. Marconi, “GDF: A Gamification Design Framework Powered by Model-Driven Engineering,” *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, 2019.

1. Gamification [↑](#footnote-ref-1)
2. Eclipse [↑](#footnote-ref-2)
3. Model Driven Software Engineering (MDSE) [↑](#footnote-ref-3)
4. Transformation [↑](#footnote-ref-4)
5. Notation [↑](#footnote-ref-5)
6. Modeling Language [↑](#footnote-ref-6)
7. MetaModel [↑](#footnote-ref-7)
8. https://www.eclipse.org/Xtext/ [↑](#footnote-ref-8)
9. https://www.eclipse.org/sirius/ [↑](#footnote-ref-9)
10. https://www.eclipse.org/atl/ [↑](#footnote-ref-10)
11. https://www.eclipse.org/acceleo/ [↑](#footnote-ref-11)
12. Mechanic-Dynamic-Aesthetic (MDA) [↑](#footnote-ref-12)
13. Aesthetics [↑](#footnote-ref-13)
14. Point [↑](#footnote-ref-14)
15. Leaderboard [↑](#footnote-ref-15)
16. Levels [↑](#footnote-ref-16)
17. Achievement System [↑](#footnote-ref-17)
18. Reward [↑](#footnote-ref-18)
19. Status [↑](#footnote-ref-19)
20. Achievement [↑](#footnote-ref-20)
21. Self-Expression [↑](#footnote-ref-21)
22. Competitions [↑](#footnote-ref-22)
23. Altruism [↑](#footnote-ref-23)
24. Submission [↑](#footnote-ref-24)
25. Motivation [↑](#footnote-ref-25)
26. Intrinsic [↑](#footnote-ref-26)
27. Extrinsic [↑](#footnote-ref-27)
28. Engagement [↑](#footnote-ref-28)
29. Instant Feedback [↑](#footnote-ref-29)
30. Collaboration [↑](#footnote-ref-30)
31. Cosentino [↑](#footnote-ref-31)
32. Do Prado [↑](#footnote-ref-32)
33. Lucr´edio [↑](#footnote-ref-33)
34. Model Driven Game Development [↑](#footnote-ref-34)
35. Jurgelaitis [↑](#footnote-ref-35)
36. Unified Process (UP) [↑](#footnote-ref-36)
37. MOODLE [↑](#footnote-ref-37)
38. Savić [↑](#footnote-ref-38)
39. Machine- Readable [↑](#footnote-ref-39)
40. Learning Objectives [↑](#footnote-ref-40)
41. Learning Resources [↑](#footnote-ref-41)
42. Instructional Strategy [↑](#footnote-ref-42)
43. E-Learning Instructional Design Meta-Language [↑](#footnote-ref-43)
44. Platform Specific Model [↑](#footnote-ref-44)
45. https://scorm.com [↑](#footnote-ref-45)
46. http://www.lamsinternational.com [↑](#footnote-ref-46)
47. http://sakaiproject.org [↑](#footnote-ref-47)
48. Arawjo [↑](#footnote-ref-48)
49. Reduct [↑](#footnote-ref-49)
50. Bucchiarone [↑](#footnote-ref-50)
51. Gamification Model Language [↑](#footnote-ref-51)
52. Game Model Language [↑](#footnote-ref-52)
53. Game Instance Model Language [↑](#footnote-ref-53)
54. Game Instance Model Language [↑](#footnote-ref-54)
55. Game Simulation Model Language [↑](#footnote-ref-55)
56. Description [↑](#footnote-ref-56)
57. Point [↑](#footnote-ref-57)
58. Badge [↑](#footnote-ref-58)
59. Coin [↑](#footnote-ref-59)