

A Feature Model Proposal for Computer Games Design

Victor T. Sarinho Antônio L. Apolinário*

State University of Feira de Santana, Brazil

Abstract

An increasingly important attribute of modern software development is that of variability. Software variability is the ability to control the software change in a particular context. To manage the game development variability, due to the game domain diversity, the most successful approach was first achieved by game engines. Nowadays, software variability can be initially identified based on the concept of feature, a logical unit of behavior that corresponds to a set of functional and quality requirements in a system. This paper presents a new way to describe a game, based on a feature modeling. The NESI model is based on four main features: Narrative, Entertainment, Simulation and Interaction. The main contribution of this paper is to propose a model to design the variability aspects of computer games, simplifying the effort of game design projects.

Keywords: software variability, feature modeling, game design.

Authors' contact:

vsarinho@gmail.com

*apolinario@comp.uefs.br

1. Introduction

According Svahnberg et al. [2005], “software variability is the ability of a software system or artefact to be efficiently extended, changed, customized or configured for use in a particular context”.

Software variability achieve many issues, “ranging from the development process itself to the various artifacts created during the development process, such as requirements specifications, design documents, source code, and executable binaries” [Svahnberg et al. 2005].

According Gurr et al. [2001], “a variability can be initially identified based on the concept of feature”. They define a feature as “a logical unit of behavior that corresponds to a set of functional and quality requirements”. Indeed, “variability management is concerned with the management of the differences between products throughout the Product Line lifecycle” [Junior 2005].

In computer games, the variability is a direct consequence of the game domain diversity, working with simulations (sports, adventure, fighting), hardware technologies (mobile games, web games), human interactions (immersion, multiplayer) and complex stories (games based on movies, RPG series) [Kent 2001].

To manage the game development variability, the most success approach was first achieved by Quake game engine [Lewis 2002], a platform based product line strategy [Bosch 2002] for game creation.

Others interesting attempts to manage the game variability, based on SPL (Software Product Line) concepts, were developed [Zhang and Jarzabek 2005; Alves 2007], but only for specific games design, and without commitment to the existing works about computer games domain [Björk and Holopainen 2005; Davidsson et al. 2004; Zagal et al. 2005; Furtado 2006; Bates and Bates 2004; Rollings and Morris 2004; Hunnicke et al. 2004; Järvinen 2007; Lemay 2007].

In this way, the objective of this paper is present a feature modeling proposal to design the variability aspects of computer games, based on the analysis of the main computer games domain engineering works.

This paper is organized as follows: Section 2 presents some important game domain engineering works. Section 3 describes the feature modeling technique used and the proposed computer game feature model. Finally, Section 4 presents the conclusions.

2. Related Work

How to make a computer game using existent components? Can a set of components be integrated or composed to produce a new computer game? What kinds of game components should exist to build a computer game? What features a software should contain to be considered a computer game? Unfortunately, there is not a direct response for these questions, but just “hints” described by game domain engineering works.

For Hunnicke et al. [2004], the Mechanics, Dynamics, and Aesthetics (MDA) framework described a game as a collection of Mechanics to compose Dynamics, and a collection of Dynamics to compose Aesthetics characteristics of a game. It was a interesting conception of game views, but without resolve the gap between game design and game development.

Björk and Holopainen [2005] described a model to support the game design using a set of game design patterns. It has been a good approach to describe the components of games and its patterns of interaction, but due to the very high level patterns description, and the repetition of semantic concepts between them, the gap between game design and game development persists.

Zagal et al. [2005] proposed the Game Ontology Project (GOP), a framework that define a hierarchy of ontology concepts: interface, rules, goals, entities, and

entity manipulation. In fact, a good representation of game rules was described, but more ontology concepts are necessary to represent game design and game development aspects.

Furtado [2006] proposed the use of a game ontology to set game implementations. Low level game design aspects were described, but focused on a specific game domain, becoming insufficient to represent the game diversity.

Järvinen [2007] presented a design methodology with psychological theories of cognition and emotions for game design, but with a great distance of game implementation aspects.

Zhang and Jarzabek [2005] proposed an RPG product line architecture (RPG-PLA), grouping similarities and differences among four RPGs (probably the first use of features on game design). As result, a specific RPG feature model, and insufficient solution to represent the game diversity, was described.

The works described above are focused on generic game domain descriptions and specific game project implementations. Therefore, the definition of an intermediate modeling technique, covering domain and design game aspects, reducing the gap between them, to describe and analyze generic game components and concepts, is necessary.

3. Computer Game Feature Model

Next subsections will present the used feature modeling technique (section 3.1), and the proposed feature models (sections 3.2, 3.3, 3.4, 3.5 and 3.6) to design computer game variability aspects.

3.1 Feature Modeling

The feature model concept was first introduced by Kang et al [1990], in the Feature Oriented Domain Analysis (FODA) method, to help the identification of important or domain special properties during analysis phase.

According Antkiewicz and Czarnecki [2004], a feature is “a system property that is relevant to some stakeholder”, and is “used to capture commonalities or discriminate among desired products”.

A feature model consists of one or more feature diagrams. These diagrams organize features into

hierarchies, as a decorated tree, containing features identified in a system.

Others notations have been proposed to expand the feature model representativeness and to provide support to different types of structural relationships. For example, Czarnecki and Eisenecker [2000] uses XOR and OR relationships to represent alternative and mutually exclusive features, and Czarnecki et al. [2005] proposes a feature cardinality-based notation.

This paper will use both feature notations to represent the feature model proposal for computer games design.

3.2 A Feature Model for Computer Games

The basis of NESI model is to define a game as a combination of four main standard features: Narrative, Entertainment, Simulation and Interaction, which was called NESI model (Figure 1).

The NESI model was based on the game conceptual definitions proposed by Esposito [2005], where a definition of videogame, combining existing researchs about game, play, interactivity, and narrative concepts, was described.

The structure of the NESI model was derived from the analysis of different domain engineering [Björk and Holopainen 2005; Davidsson et al. 2004; Zagal et al. 2005; Furtado 2006; Bates and Bates 2004; Rollings and Morris 2004; Hunicke et al. 2004; Järvinen 2007; Lemay 2007] and implementation works [Zhang and Jarzabek 2005; Alves 2007; Trinta 2007] of computer games.

A *Narrative* is a Flow, a dynamic script trying to achieve Goals following some defined game Rules. *Entertainment* is represented by the player Immersion during the game, accompanied by a Theme proposal, to characterize the player and the game in the proposed reality. *Simulation* is a combination of Elements (resources for play) and their Relationships, that occurs in a defined Environment (spaces to play). The human *Interaction* is represented by the Control (game input) and the Presentation (game output) features, consolidated by the Gameplay feature that manages the game state and execution as a whole.

To illustrate the proposed model, just imagine a Chess game. Chess is a *Narrative*, trying to execute some steps in a Flow (prepare the game, start the game, wait for player movement), according defined Rules

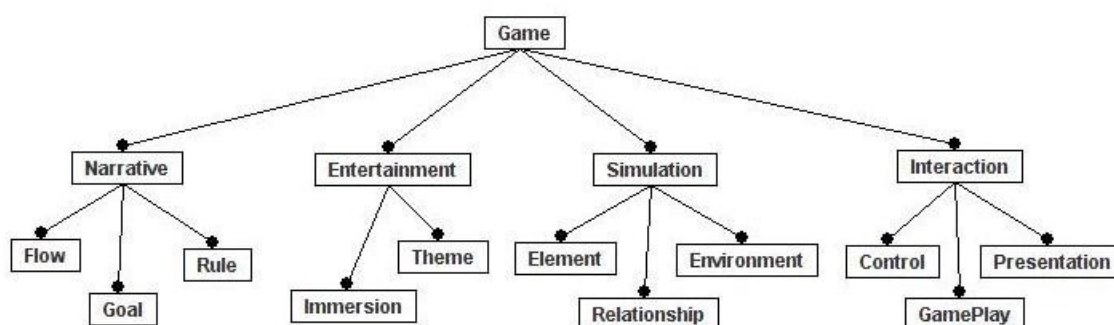


Figure 1: NESI model for computer games representation.

(types of movement) to complete these Goals (take the queen, kill the king). Chess is a kind of *Entertainment* because it offers a combination of Immersion feelings (Emotion, Challenge), following a Theme (war of kingdoms) according a Story or not (kill the bad white king for example) and using some Avatars (queen, bishop, king). Chess is a *Simulation*, by the representation of Environment (game board), Elements (white and black pieces), and their Relationships (Alliances in a multiplayer challenge, piece Consumption). Finally, Chess is a *Interaction*, because it offers Presentation opportunities by Visual (3D and 2D Cameras to show the Chess board) and Aural (special SoundEffects for each player movement) features. To Control the Chess game, an IndirectControl (PointAndClick using the mouse to move any piece) is more used, and for the Gameplay, a Persistent Context (storing the game state) with TurnBased Policy Execution (one player movement each time) and PreGame Customization (config the game before play it) are the most common used features.

3.3 Narrative Features

As illustrated in Figure 1, the Narrative feature is composed by three subfeatures: Flow, Goal and Rule.

The Flow feature, which represents a sequence of operations [Taylor 2006; Brown 2006], is basically a collection of Events and Nodes (Figure 2).

Each Node contains: a collection of ActionList, which contains Actions to be performed; and a

collection of Transitions, to define the new execution Node in the Flow when one Transition is performed.

During the game execution Events are used to distinguish what ActionLists or Transitions will be performed. As result, for each Event in the execution Node, new Actions or Transitions can be performed, triggering new Events again.

Different types of Nodes could be used: Fork, Join,

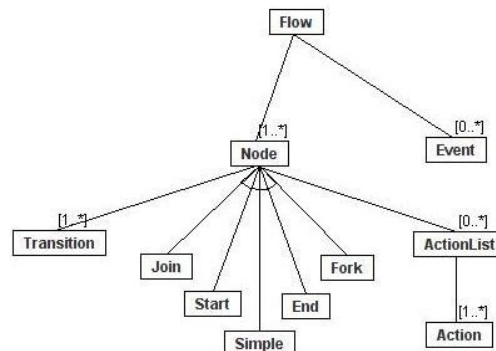


Figure 2: Flow feature diagram.

Start, and others. When these nodes are combined they compose a Flow in the game.

According Zagal et al. [2005], “goals are the in-game objectives or conditions that the player must meet if he expects to succeed at the game”.

To complete a game, goals must be attained. The Goal feature (Figure 3) is composed by many SubGoals, which contains the following features: Change (the Goal is Static, Dynamic or can be Customized by the player), Constraint (regular expression composed by and, or, not and sequence

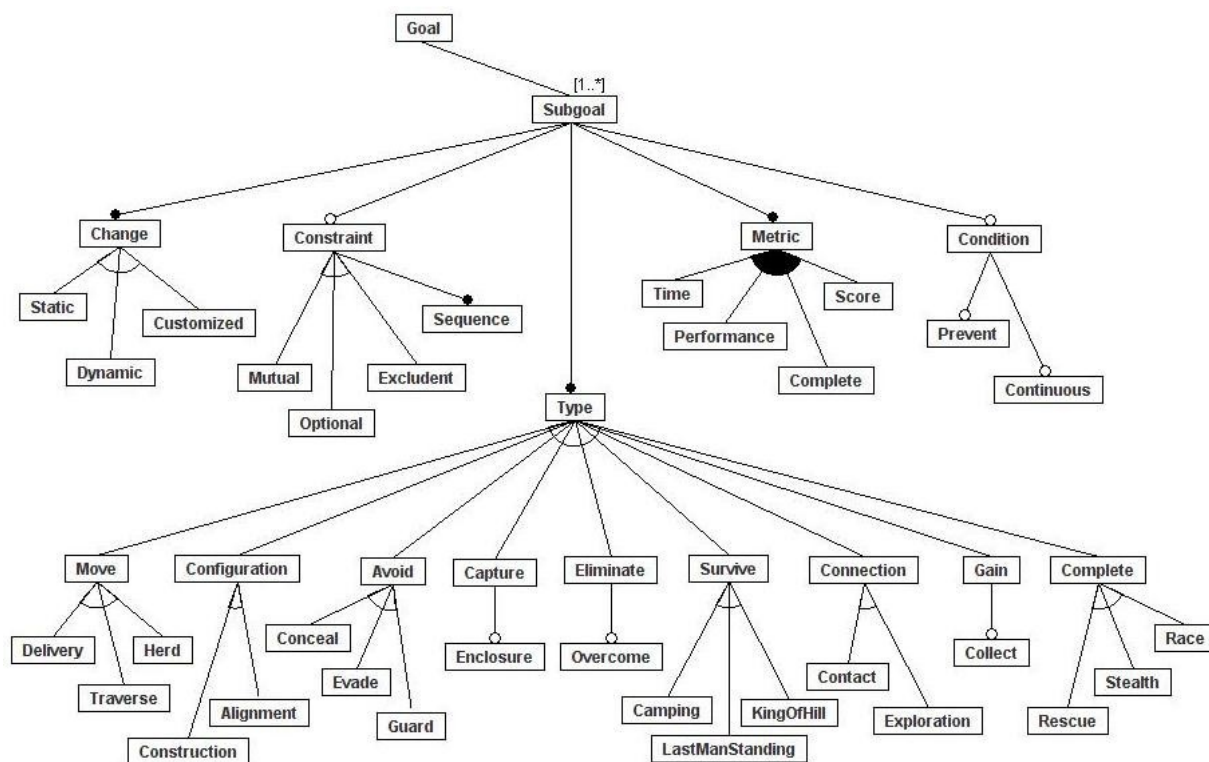


Figure 3: Goal feature diagram.

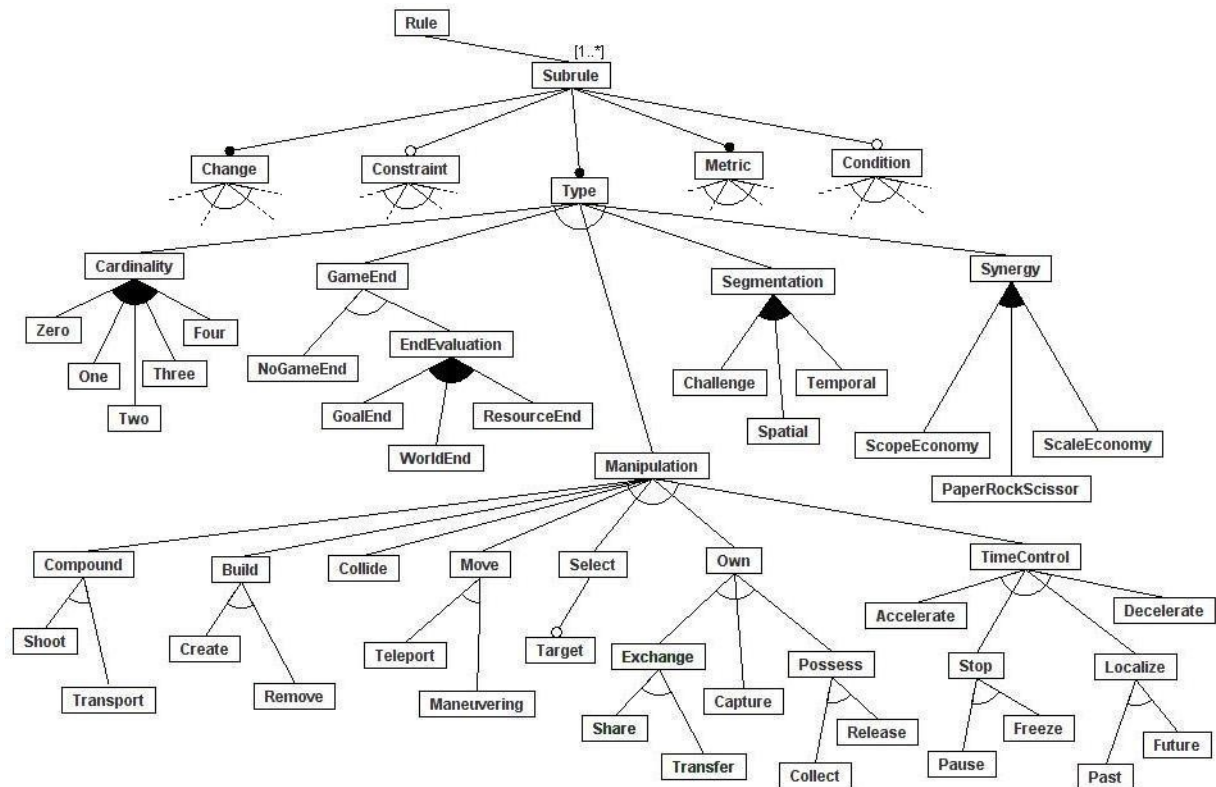


Figure 4: Rule feature diagram.

rules), Type (different kinds of goals in a game), Metric (modes of evaluation of a goal progress) and Condition (the goal is Prevent a goal complete, maintain a Continuous state for a period to conclude the goal [Björk and Holopainen 2005]).

Similar to Goal, a Rule can be considered as a game condition, a game constraint which must be validated in the game execution to perform game entity manipulations.

Figure 4 shows the different types of rules in a game, which are subdivided in: Cardinality, GameEnd, Manipulation, Segmentation and Synergy. SubRule, Change, Constraint, Metric and Condition, similar to Goal feature (with the same behaviour), are subfeatures of the Rule feature too.

According Zagal et al. [2005], the Cardinality of gameplay refers to the “degrees of freedom the player has with respects to movement (or the control of movement) in a certain game”. Zero, One, Two, Three and Four cardinalities are proposed in the feature model, that can be found, respectively, in the following game examples: Guitar Hero, Space Invaders, Pac-Man, Quake and Neko.

GameEnd describes some ways to determine if a game ended or not. GoalEnd (goal completed), ResourceEnd (consumption of all resources), WorldEnd (end of the world was achieved), and NoGameEnd (Sims, Second Life, for example) are shown in the game feature model, acting as initial proposals to finish the game play, and allowing new different possibilities to determine the game end.

Manipulation validates attempts to change the context of entities in the game. Create, Share, Capture,

Target, Shoot, Teleport, and others can be considered as a Manipulation feature.

According Zagal et al. [2005], the Segmentation of gameplay describes “how a game is broken down into smaller/shorter elements or chunks of gameplay”. One Segmentation rule may present simultaneously: Challenge (missions, boss stage), Spatial (sectors in a build) and Temporal (time limit to explore a sector) features.

Synergy defines the relationships between rules, describing strategic options available to players [Zagal et al. 2005]. ScaleEconomy, ScopeEconomy and PaperRockScissor describes rules constraints for quantity, quality and transitive effects between entities, respectively.

3.4 Entertainment Features

Composed by Immersion and Theme features, the Entertainment feature define entities to response the following questions: What makes a game fun? What features are necessary to build a funny game?

For Immersion, Björk and Holopainen [2005] describe that “games require the player attention, and as such can make players focus on gameplay to the extent that they feel immersed in the games”. This Immersion can take many forms, depending on “what type of activity the players are performing, putting them deeply focused on the interaction within the game”.

The Immersion feature (Figure 5) in the NESI model proposes four types of immersion features: Cognitive, Emotional, Spatial and Challenge.

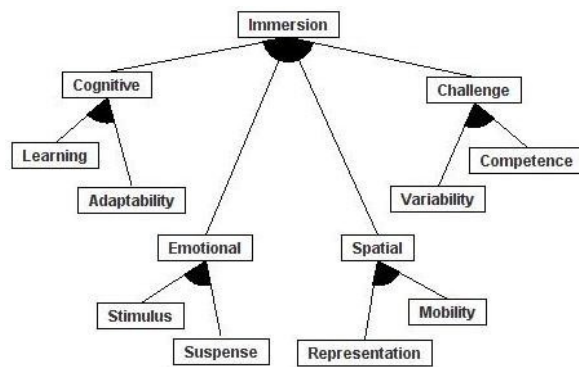


Figure 5: Immersion feature diagram.

Cognitive presents Learning opportunities to promote memorizing in a constructive play for puzzle solving, and Adaptability to reach the right level of complexity in a game [Björk and Holopainen 2005].

Emotional describes “Suspense as modulation of hopes and fears through elements of uncertainty”, and Stimulus because “games privilege so-called prospect-based emotions that are always focusing on events and their outcomes” [Järvinen 2007].

Spatial presents a real and contextualized game entity Representation followed by the player Mobility inside them [Björk and Holopainen 2005].

In *Challenge*, a game must explore the player Competence, offering a game Variability by new and continuously player adventures [Lemay 2007].

According Järvinen [2007], a Theme is the subject matter of the game, is a collection of functions to represent a metaphor for the system and the proposed rule set.

In the NESI model, a Theme feature (Figure 6) is basically composed by Avatars and a Story (which is optional). A Story is a collection of Acts with a time Momentum (story Beginning, Middle or End) for each one [Bates and Bates 2004]. Each Act is formed by one or more Types, which can be a ScriptEvent (a dynamic Act based on the player choices during the Story course), a CutScene, a BackStory, a Dialog or a

Surprise (as result of Secrets or SideEffects during the game).

An Avatar is an actor in the game, and is subdivided in Agent (“game entities with intentional behavior” [Zagal et al. 2005]) and Player (“those who play, in various formations and with various motivations, by performing game mechanics in order to attain goals” [Järvinen 2007]). Each Avatar has many Skills (Luck, Dexterity and others) independent if it is a Player or an Agent. According Björk and Holopainen [2005], Skills are the “numerical representation of how likely an avatar is to succeed with an action, and what possible consequences the action has”.

An Agent can also represent a Mule (“character that is set, typically by using scripts, to perform long, monotonous and specialized sets of actions” [Björk and Holopainen 2005]) or an Enemy (“character that hinder the players trying to complete the goals” [Björk and Holopainen 2005]), and an Enemy could be a Boss (“a more powerful enemy the players have to overcome to reach certain goals in the game” [Björk and Holopainen 2005]).

3.5 Simulation Features

According Narayanasamy et al. [2006], “the advanced computational capabilities in modern personal computers have made it possible for consumers to experience simulations with a high degree of verisimilitude through simulation games (a.k.a. Sims)”.

A good example is the serious games category, a result of applying simulation technology to nonentertainment purposes (mostly training), contributing to the technology exchange between simulation games and simulators [Narayanasamy et al. 2006].

Other game categories uses simulation concepts too, when they abstract real life entities and concepts to gain more veracity in these game contexts.

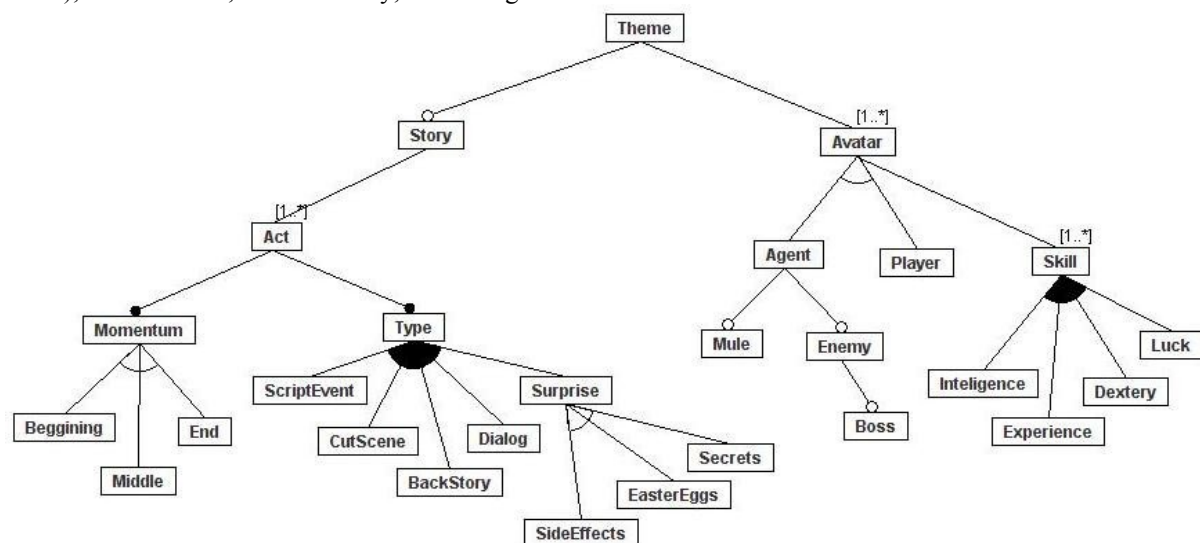


Figure 6: Theme feature diagram.

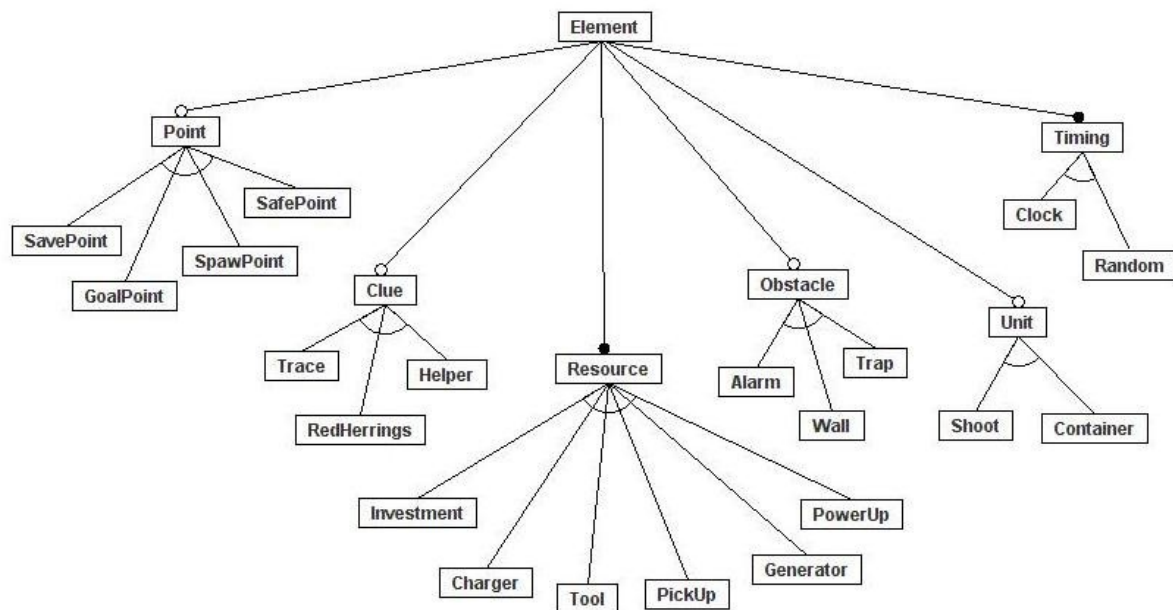


Figure 7: Element feature diagram.

So, is impossible to think in a game without simulation features. For this purpose, the NESI model presents a Simulation feature, composed by Element, Environment and Relationship main features.

The Element feature (Figure 7) represents the structural game aspects, and contains the following features: Point, Clue, Resource, Obstacle, Unit and Timing.

Points are “specific places or moments in a game” [Zagal et al. 2005]. The NESI model proposes some Points to save the game (SavePoint), to stay in secure mode (SafePoint), to complete a game goal

(GoalPoint), and to generate new game entities (SpawnPoint).

Clues are “game elements that give the players information about how the goals of the game can be reached” [Björk and Holopainen 2005]. For this, Traces (soldiers traces on a terrain, for example), RedHerrings and Helpers (villagers and their stories, for example) features are proposed. Bates and Bates [2004] disagrees with the inclusion of RedHerrings in a game, because they could harm in some moment the player Entertainment, and the player already has enough problems to worry about during the game. However, it is a characteristic that can be implemented

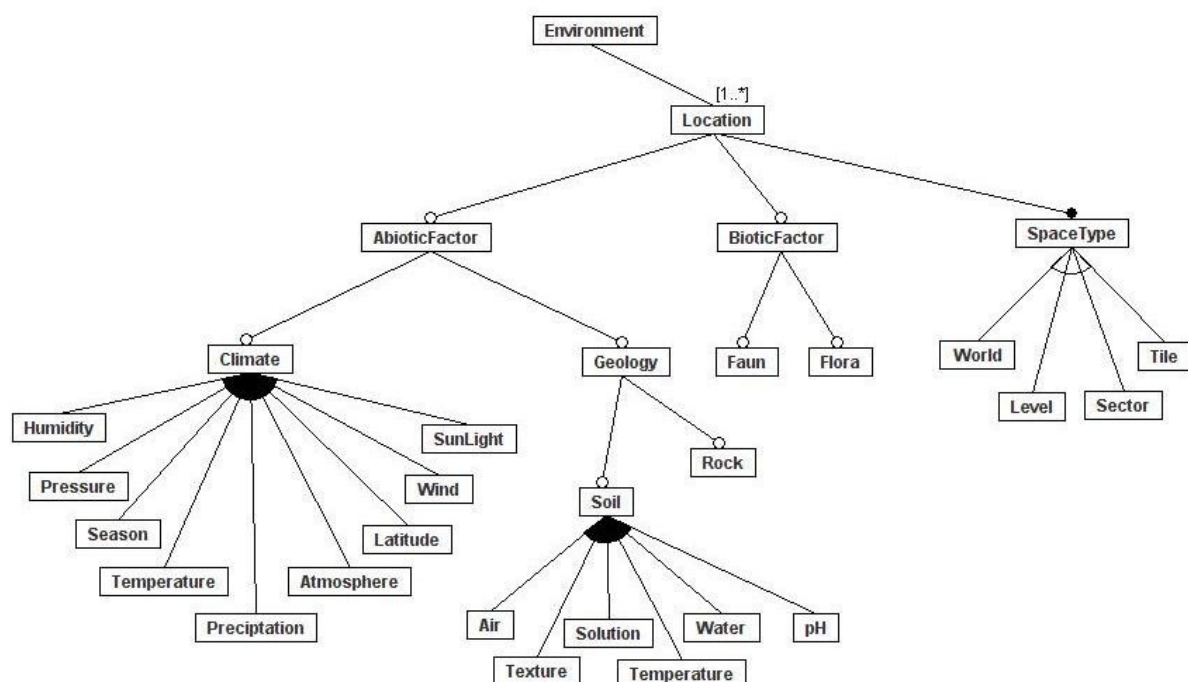


Figure 8: Environment feature diagram.

in the games, so it should be described in the proposed model.

Resources are “game elements that are used by players to enable actions in a game” [Björk and Holopainen 2005]. Some proposed resources are: Investment (in order to reap the rewards later), Charger (speed boosters for example), Tools (weapons, armor, gadgets for example), PickUp (elements that can be collected by players), Generator (mechanisms that bring game elements to the game) and PowerUp (power pill in Pac-Man for example).

Obstacles are “game elements that hinder the players from taking the shortest route between two places” [Björk and Holopainen 2005]. Alarms, Walls and Traps are some possible and proposed Obstacle features.

Units are “groups of game elements under the player's control that let the player perform actions to influence the Game World” [Björk and Holopainen 2005]. Shoot (performs multiple actions during the release and collision of the projectile object) and Container (game element that can store other game elements) features are defined.

Timing are game elements to manage time events which can or cannot be exactly predicted in the game. Clock (synchronous events) and Random (asynchronous events) features are proposed.

The Environment feature (Figure 8), representing the game space for play [Järvinen 2007], is composed by many Locations. Each Location can take a different SpaceType (World, Level, Sector, Tile and others), and could represent different Biotic and Abiotic factors available on specific game Locations.

For AbioticFactors, Climate (with Humidity, Pressure, Temperature, Precipitation and others) and Geology (with Soil and your parameters: pH, Air, Water, Texture, etc.; and Rocks) aspects are the most outstanding features.

For BioticFactors, Faun (types of species) and Flora (types of vegetation) features in a location, with its rich diversity, can be represented.

Relationship feature (Figure 9) defines the interaction type between Element-Element, Element-Environment and Environment-Element. For that,

Biological (essential for life support) and Social (essential for communities support) interactions are expected.

Biological features describes: Consumption (Parasitism, Predation, Herbivory and others), Competition (struggle to achieve a certain goal) and Cooperation (in a biological level, like Commensalism and Mutualism) features.

Social features describes: Communication (traditional chat between players), Collaboration (“players cooperate to reach goals or subgoals of the game” [Björk and Holopainen 2005]), Solidarity (players share Penalties and Rewards between them) and Negotiation (“a situation where the players confer with each other in order to reach an agreement or settlement” [Björk and Holopainen 2005]).

It is important to reinforce the game designer responsibility to choice the Relationship type between Elements and Environment, due to conceptual Relationship similarities that could confuse the designer (Cooperation and Collaboration for example).

3.6 Interaction Features

According Zagal et al. [2005], to create an interactive experience in games, it must provide interesting input and output resources.

Trinta [2007] describes interaction as “how the player performs its actions in the virtual world, which is also dependent on the context of implementation and the perception of the player at any given time”.

The NESI model proposes three features to achieve the desired game Interaction: Presentation (managing sensory outputs for the player), Control (managing players actions) and Gameplay (managing all created GameSessions necessary to start the Gameplay).

Zagal et al. [2005] described that games present themselves to the player through the senses, through multiple sensory channels or focused on only one.

In this way, Presentation feature is composed by three sensory features: Visual (most important Presentation game characteristic), Aural (representing every sound that can be reproduced during the game) and Tactil (with few game domain studies).

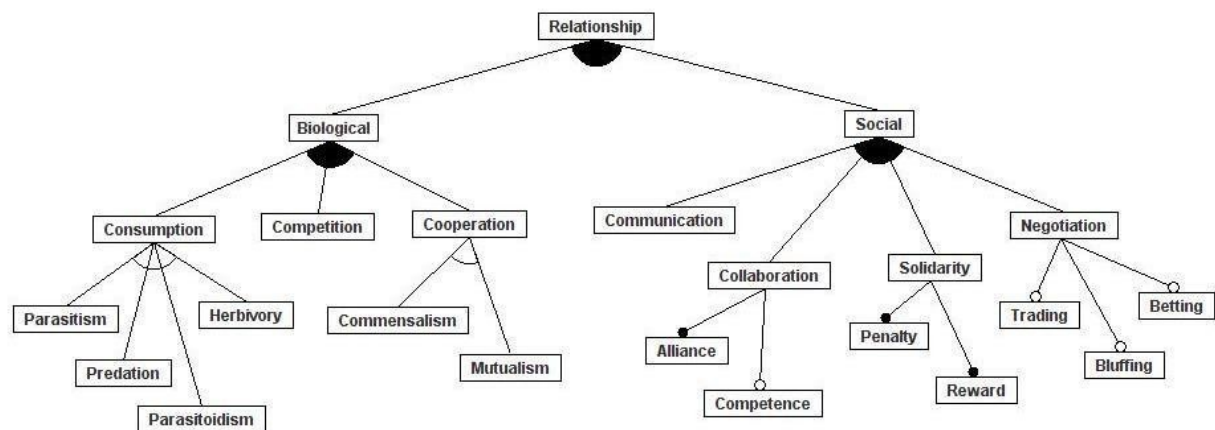


Figure 9: Relationship feature diagram.

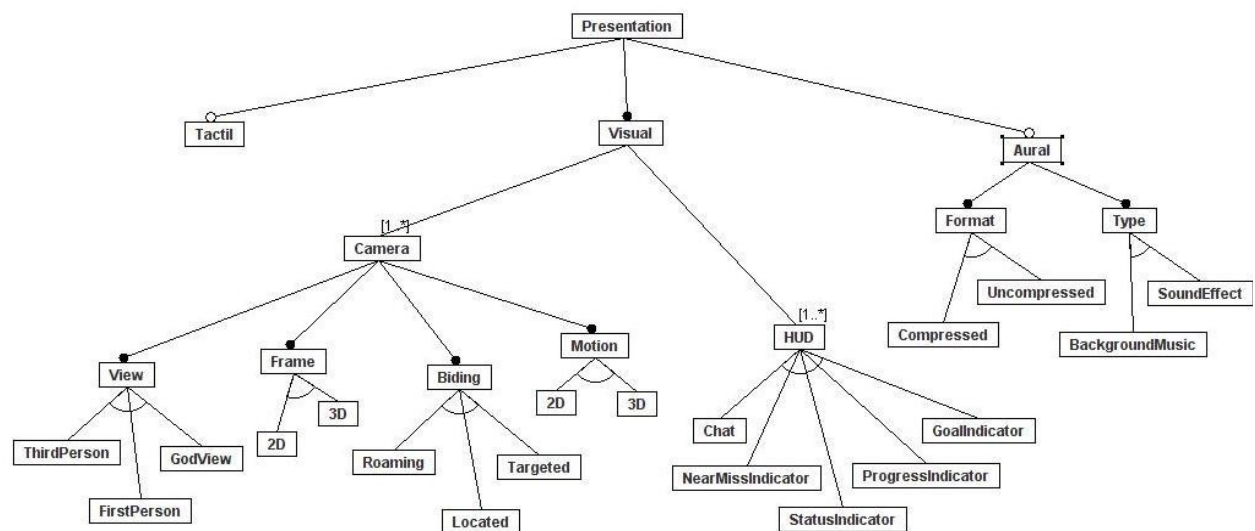


Figure 10: Presentation feature diagram.

For the NESI model, the Visual game aspect is formed by a combination of Cameras and HUDs (Head-Up Displays) features.

According Zagal et al. [2005], a Camera is “a metaphor to describe a graphical representation of the game world in which the world, the entities within it and their spatial relationships are depicted visually”.

To represent this metaphor, Camera feature is composed by the following features: View (perspective from which the player perceive and interact with the space), Frame (how the camera represents the space), Biding (maintain continuity in the view), and Motion (how the camera moves within or in relation to the represented space) [Zagal et al. 2005].

The HUD objective is provide to the game player some information about the game in general, such as the lives indicator, health indicator, time indicator and others [Furtado 2006]. In summary, your goal is to indicate progress (ProgressIndicator), status (StatusIndicator), communication (Chat), and proximity for a goal (GoalIndicator) or to lose it (NearMissIndicator).

According Zagal et al. [2005], an Aural output varies over its constituent wavelengths, like a BackgroundMusic or SoundEffect to reflect the dynamical game state. To represent an Aural feature, a sound Type (SoundEffect or BackgroundMusic) and Format (Compressed or Uncompressed) are expected.

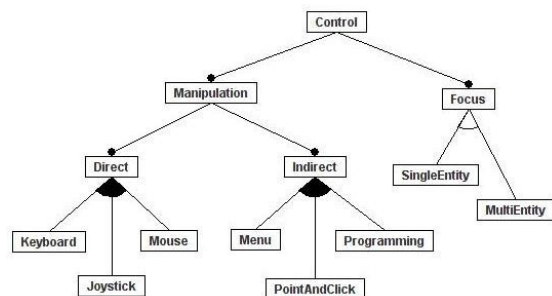


Figure 11: Control feature diagram.

For game control objectives, “a game can use multiple approaches to its control mapping as well as

multiple approaches for the same control” [Zagal et al. 2005]. Indeed, some games provides to the player a level of Direct control over his Avatars actions, and other games take an Indirect control approach, using either menus or point and click interfaces [Zagal et al. 2005].

In this way, Figure 11 shows the proposed Control feature, composed by Direct and Indirect player commands, and the Focus of manipulation (Single or Multiple game entities). As shown, at least one type of command to manage the game play is expected in a game.

According Björk and Holopainen [2005], a game instance, represented as Gameplay feature (Figure 12), is the “whole lifetime of the game”, and a GameSession is the “whole activity of a player playing one game”.

Therefore, Gameplay contains a collection of GameSessions, which are composed by collections of Context, Execution and Customization features.

Game Context is defined as “where, when, and why the gaming encounter takes place” [Järvinen 2007]. For Trinta [2007], “all relevant information to represent the situation of an entity that composes or interacts with the application” is called Context.

In the NESI model, the game Context is a collection of entity Instances, with a Persistent status capability. Some games could include the ClosurePoints feature, “points where most of the information about game elements and actions performed become irrelevant and are discarded” [Björk and Holopainen 2005], a game point without return.

As an Instance can be of any game entity type, this feature is composed by NarrativeInstance, EntertainmentInstance, SimulationInstance and InteractionInstance features.

An Execution is equivalent to a game thread, necessary to execute actions, generated from players and multiplayers (agents or not), in a parallel way.

Each Execution has a game Policy that determines the player command execution at a given time [Trinta 2007]. TurnBased (“only one player or agent may

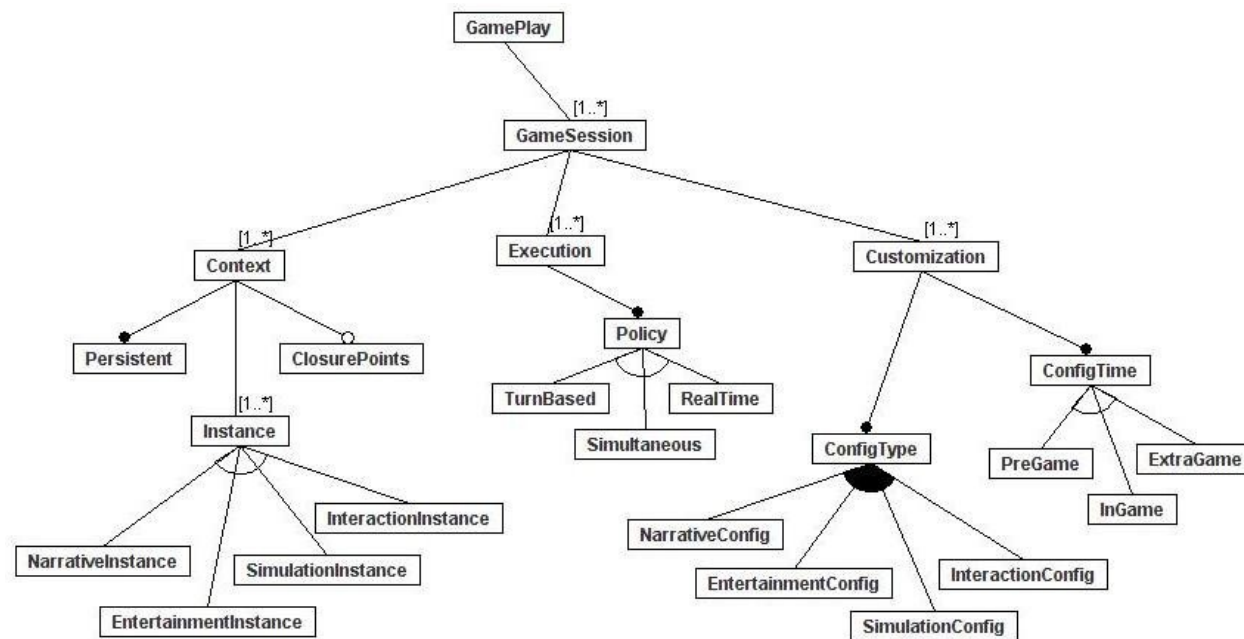


Figure 12: GamePlay feature diagram.

perform activities in the game at a given moment” [Zagal et al. 2005]), RealTime (“do not require player actions to change the game state” [Björk and Holopainen 2005]), and Simultaneous (after receive all next player commands, the game state will change [Trinta 2007]) policies are the most common used.

According Zagal et al. [2005], customize an entity means basically “the ability of the player to create or modify it according to his/her taste”. This customization can be performed in a number of ways and involve several aspects of the affected entity, such as graphics, sounds, name, performance, and others.

In the NESI model, the Customization feature is composed by two features: ConfigType and ConfigTime.

ConfigType represents the customization of all game entity features. Therefore, the NarrativeConfig, EntertainmentConfig, SimulationConfig and InteractionConfig features represent all possible game entity configurations.

According Zagal et al. [2005], ConfigTime can be classified as: PreGame (not performed where the game action takes place), InGame (performed by the player where the game action actually takes place) and ExtraGame (achieved through the use of software or tools external to the game itself).

4. Conclusion

This paper presents the NESI feature model which describes game modeling aspects based on Narrative, Entertainment, Simulation and Interaction features.

As the first collection of feature models for generic games design, the great importance of this work is reduce the gap between generic game domain descriptions and specific game project implementations. This goal is achieved organizing different game domains studies in a more formal, and domain level, design approach.

Using the NESI model, the diversity and variability involved in the game modeling, and treated by the game designer, can be better controlled.

This is the result of the simplification of the game domain to four main feature aspects, allowing a more objective decision about what feature aspects a final game project will contain.

A more systematic and simplified game reuse approach is obtained too, because the NESI model can be used to guide the decomposition of existing games (creating new core assets for future game projects), and to develop new games from the ground up (configuring available core assets that implement game features).

The NESI model has been thought to be independent of graphics, audio, physics, artificial intelligence, or other type of game engine. It focus on the game domain, on the g-factor (game factor) inside the game project [BinSubaih and Maddock 2008].

For future works, a game product line and a game domain engine, based on the NESI model, and the integration of these techniques to some existent game engines, using a game software bus proposal, will be developed. Case studies showing how the model could be used in the decomposition of existing games and in the development of new game projects will be described too.

References

- ALVES, V. R., 2007. *Implementing Software Product Line Adoption Strategies*. PhD thesis, Federal University of Pernambuco.
- ANTKIEWICZ, M. AND CZARNECKI, K., 2004. FeaturePlugin: feature modeling plug-in for Eclipse. In: *Proceedings of the 2004 OOPSLA workshop on eclipse technology eXchange, October 24th to 28th, 2004, Vancouver, Canada*, pp 67-72. New York, USA, ACM Press.

- BATES, R. AND BATES, B., 2004. Game Design, Second Edition. *Thomson Course Technology*, ISBN: 1592004938, 9781592004935.
- BINSUBAIH, A. AND MADDOCK, S., 2008. Game Portability Using a Service-Oriented Approach. *International Journal of Computer Games Technology, Volume 2008, Article ID 378485, 7 pages*. Hindawi Publishing Corporation, doi:10.1155/2008/378485.
- BJÖRK, S. AND HOLOPAINEN, J., 2005. Patterns in Game Design. Charles River Media, ISBN: 1-58450-354-8.
- BOSCH, J., 2002. Maturity and Evolution in Software Product Lines: Approaches, Artefacts and Organization. In: *Proceedings of the Second Conference Software Product Line Conference (SPLC2)*, pp. 257-271.
- BROWN, R., LIM, A., WONG, Y., HENG, S. AND WALLACE, D., 2006. Gameplay Workflow: a Distributed Game Control Approach. In: *CyberGames '06: Proceedings of the 2006 international conference on Game research and development*, pp. 207-214. Australia, Perth, Murdoch University, ISBN:86905-901-7.
- CZARNECKI, K. AND EISENECKER, U. W., 2000. Generative Programming: Methods, Tools, and Applications. Addison-Wesley.
- CZARNECKI, K., HENSEN, S. AND EISENECKER, U. W., 2005. Formalizing cardinality-based feature models and their specialization. *Software Process Improvement and Practice*, 10(1):7-29, jan/mar.
- DAVIDSSON, O., PEITZ, J. AND BJÖRK, S., 2004. Game Design Patterns for Mobile Games. *Project report to Nokia Research Center, Finland*. Available from: http://procyon.lunarpages.com/~gamed3/docs/Game_Design_Patterns_for_Mobile_Games.pdf [Accessed 05 August 2008].
- ESPOSITO, N., 2005. A Short and Simple Definition of What a Videogame Is. In: *Proceedings of the Digital Interactive Games Research Association Conference (DiGRA 2005), Vancouver B.C., June, 2005*, pp. 2-2.
- FURTADO, A. W. B., 2006. Defining and Using Ontologies as Input for Game Software Factories. In: *Proceedings of the 3rd Brazilian Symposium on Computer Games and Digital Entertainment*.
- GURP, J. V., SVAHNBERG, M. AND BOSCH, J., 2001. On the Notion of Variability in Software Product Lines. In: *Proceedings of the Working IEEE/IFIP Conference on Software Architecture (WICSA), August 2001*, pp. 45-55.
- HUNICKE, R., LEBLANC, M., AND ZUBECK, R., 2004. MDA: A formal approach to game design and game research. In: *Proceedings of the AAAI-04 Workshop on Challenges in Game AI, July 2004*, pp. 1-5.
- JARVINEN, A., 2007. Introducing Applied Ludology: Hands-on Methods for Game Studies. In: *Proceedings of the DiGRA 2007 Situated Play. International Conference of the Digital Games Research Association, September 24th to 28th, 2007, Tokyo, Japan*, pp 134-144.
- JUNIOR, E. A. O., GIMENES, I. M. S., HUZITA, E. H. M. AND MALDONADO, J. C., 2005. A Variability Management Process for Software Product Lines. In: *CASCON '05: Proceedings of the conference of the Centre for Advanced Studies on Collaborative research, Toronto, Ontario, Canada, pp 225-241*. IBM Press.
- KANG, K., COHEN, S., HESS, J., NOVAK, W. AND PETERSON, S., 1990. Feature-Oriented Domain Analysis (FODA): Feasibility Study. *CMU/SEI-90-TR-21, SEI, USA*.
- KENT, S. L., 2001. The Ultimate History of Video Games. *Tree Rivers Press*, ISBN:0761536434.
- LEMAY, P., 2007. Developing a pattern language for flow experiences in video games. In: *Proceedings of the DiGRA 2007 Situated Play. International Conference of the Digital Games Research Association, September 24th to 28th, 2007, Tokyo, Japan*.
- LEWIS, M. AND JACOBSON, J., 2002. Games engines in scientific research. *Communications of the ACM*, vol. 45, no. 1, pp. 27-31.
- NARAYANASAMY, V., WONG, K. W., FUNG, C. C. AND RAI, S., 2006. Distinguishing Games and Simulation Games from Simulators. *ACM Computers in Entertainment, Vol. 4, No. 2, April-June 2006*, pp. 1-18. New York: ACM Press, ISSN:1544-3574.
- ROLLINGS, A. AND MORRIS, D., 2004. Game Architecture and Design : A New Edition. *New Riders Publishing*, ISBN: 0735713634.
- SVAHNBERG, M., GURP, J. V. AND BOSCH, J., 2005. A Taxonomy of Variability Realization Techniques. *Software—Practice & Experience, Volume 35, Issue 8 (July 2005)*, 705-754. New York: John Wiley & Sons, ISSN:0038-0644.
- TAYLOR, M. J. GRESTY, D. AND BASKETT, M., 2006. Computer game-flow design. *ACM Computers in Entertainment, Vol. 4, No. 1, January 2006*, pp. 5. New York: ACM Press, ISSN:1544-3574.
- TRINTA, F. A. M., 2007. *Definindo e Provendo Serviços de Suporte à Jogos Multiusuário e Multiplataforma: Rumo à Pervasividade*. PhD thesis, Federal University of Pernambuco.
- ZAGAL, J., MATEAS, M., FERNANDEZ-VARA, C., HOCHHALTER, B. AND LICHTI, N., 2005. Towards an Ontological Language for Game Analysis. In: *Proceedings of the Digital Interactive Games Research Association Conference (DiGRA 2005), Vancouver B.C., June, 2005*, pp. 3-14.
- ZHANG, W. AND JARZABEK, S., 2005. Reuse without Compromising Performance: Experience from RPG Software Product Line for Mobile Devices. In: *Proceedings of the 9th Int. Software Product Line Conf., SPLC'05, Sept. 2005, Rennes, France*, pp. 57-69.