Separating the Autonomous Behaviors and Coordination Regimes of Non-Player Characters

Gonzalo Milla-Millán and Juan Fdez-Olivares

Dpto. de Ciencias de la Computación e I.A., Universidad de Granada, Spain
{gmillamillan,faro}@decsai.ugr.es

Abstract. Authoring believable Non-Player Characters (NPCs) - which is a bottleneck in the development process - and generating interesting and varied stories where they interact - which is a key aspect to enrich the user's experience - are major concerns in Interactive Storytelling (IS). In this paper we propose an authoring framework that separates the autonomous behaviors of each NPC (that is, its basic capabilities and their feasible combinations) from the coordination regimes that drive the interactions among characters, thus facilitating the reuse of knowledge in order to reduce the authoring effort. A decentralized multi-agent narrative engine is also presented, which uses this encapsulated knowledge to provide story variability, simulating several actors and a performance director who collaborate to build the story. An implementation using Hierarchical Task Networks (HTN) planning is described.

1 Introduction

Authoring believable NPCs and generating interesting and varied narratives where they interact in a common virtual world are major concerns in IS. The variability of the story, which is a key aspect to enrich the user's experience, depends directly on the *autonomous behaviors* of each NPC (that is, the basic capabilities that she can perform by herself and their feasible combinations) and the *coordination regimes* that drive the interactions among characters. For instance, if two NPCs (Jerry and George) are to leave a room through a door that only fits one person through at a time, at least three different narratives with dramatic value can be generated: (1) Jerry leaves before George, (2) George leaves before Jerry, and (3) Jerry and George collide at the door while trying to cross it at the same time. The autonomous behaviors of each NPC (to walk to the door and cross it) in these three narratives are equal, so the only thing that has changed is the applied coordination regime. Therefore, separating the authoring of the autonomous behaviors from the coordination regimes will help to provide story variability. Furthermore, this separation can also help in making the authored knowledge reusable across different scenarios and even IS systems, which can reduce the required authoring efforts.

Automated planning is one of the predominant technologies in IS, due to the good correlation that exists between the causal structure of plans and the constructs that have been associated with narrative reasoning [17]. Regarding the authoring process, most of the planning approaches in IS can be divided in *plot-based* [17, 16, 12] and *character-based* [5, 2, 10, 15]. Plot-based approaches set the focus on modeling the events of the story, thus making the autonomous behaviors of the NPCs and their coordination regimes to be intertwined in the planning domain. Alternatively, character-based approaches set the focus of the authoring process in the characters, which seems to be better suited for IS due to its better integration with the interacting user [1]. In character-based approaches, the interactions among NPCs are either mixed with their autonomous behaviors in the planning domains [5, 2, 10], or directly encoded as total order constraints in a (task-oriented) global planning goal that represents the abstract story to be told [15]. Therefore, no specific distinction between autonomous behaviors and coordination regime is managed in any of these approaches.

There are recent works in the literature that aim to modularize the authoring process, trying to encapsulate some aspects like the different nuances on the way that NPCs perform actions [19], the evolution of social relationships among NPCs due to the social interactions between them [13], the expectations by a reference entity such as the non-written ethic rules of a city [14] and the beliefs of each character about the others [11]. However, all these approaches are focused on the social nature of NPCs, which is something internal to them and orthogonal to the interactions that may arise due to their placement in the same *physical* virtual world (such as two NPCs simultaneously trying to use a non-sharable resource).

On the other hand, several attempts can be found in the Multi-Agent Systems literature to encapsulate coordination regimes. Von Martial presents in [21] a multi-dimensional taxonomy of interactions among agents and proposes petri nets as a way to represent plans and coordinate them, but no formal process is specified for coordination management. Castellfranchi et al try to advance toward the same objective in [3], focusing on the concepts of *dependence* and *influence* among agents. This work is theoretical and does not reflect any implementation nor ideas about how this approach could be embedded in a planning system. Smith et al provide a quite detailed and interesting ontology of interactions in [20], but the coordination regimes are applied at execution time, taking into account only the set of executable ground actions, which makes this coordination approach not directly applicable for planning purposes in IS.

In this work, we propose an authoring framework that separates the autonomous behaviors of NPCs and the coordination regimes that drive their interactions in the common physical world where they are placed (section 3). This separation allows to work quite independently in both authoring aspects, facilitating the reuse of knowledge. A decentralized multi-agent narrative engine is also presented (section 4), which allows for story variability by dynamically selecting and merging this knowledge. An implementation is shown (section 5), where authoring is done using a formal planning language and the narrative engine follows a Multi-Agent Planning (MAP) approach. The paper finishes with some reflections about future work (section 6). But first of all, we motivate this work in the next section through an example extracted from the literature.

2 A Motivating Example

The following example of coordinated activities, which can be seen as a little narrative where two NPCs interact, is extracted from [20]:

> Jerry and George want to leave a room, and so they independently walk towards the door, which can only fit one person through at a time. Jerry graciously permits George to leave first.

Though this example is minimalistic, it contains all the authoring elements of interest for this paper. Concretely, this narrative can be broken down into the following elements: (1) a *common virtual world* where both NPCs are placed: a room with a door; (2) two *autonomous NPCs*: Jerry and George; (3) a set of basic *capabilities* for each NPC (the basic actions that are directly performable in the virtual world): to walk to the door, to cross the door; (4) an *autonomous behavior* for each NPC to achieve her narrative goal, which is composed by a partially ordered set of her capabilities: *first* walk to the door, *and then* cross the door[1]; and (5) a *coordination regime* which determines how the intended behaviors of each NPC are coordinated: given by the sentences *"... the door, which can only fit one person through at a time ..."* (which implies an interdependency between the `cross-door` capability of each NPC) and *"... graciously permits ... to leave first"* (which suggests a priority among character behaviors).

If these elements can be authored in a modularized manner, it will facilitate the reuse of the authored knowledge. Furthermore, changing any of these authoring elements (e.g., using a different coordination regime) might originate a different narrative, so modularization also favors story variability. For instance, at least the three narratives mentioned above - (1) Jerry leaves before George, (2) George leaves before Jerry, and (3) Jerry and George collide at the door while trying to cross it at the same time - can be obtained by simply changing the coordination regime and letting the rest of elements unaltered.

In the next section we propose an authoring framework that allows for this modularization by encapsulating the authoring elements explained above.

3 Authoring Framework

Figure 1 depicts the proposed authoring framework where the authoring elements are separated in the *Physical Aspects* of the virtual world, the *Autonomous Behaviors* of NPCs and the *Coordination Regimes* among them.

---

[1] Though in this case the autonomous behaviors and basic capabilities for both NPCs are the same, they could differ. For instance, if the backstory of Jerry and George states that both of them work for a journal where Jerry is employed as a journalist and George as a secretary, Jerry will have the capability of writing an article (which George has not) and George will have the capability of updating the director's agenda (which Jerry has not). Also, both NPCs would have private autonomous behaviors that make use of their private capabilities.

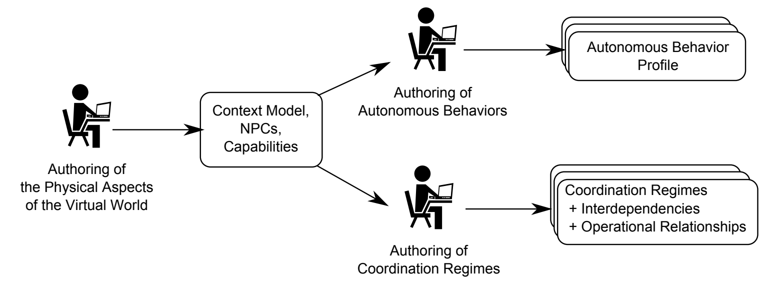

Fig. 1. Authoring Framework.

The physical aspects of the virtual world include all the elements that represent the virtual world of the narrative and the elements that can alter it, namely: (1) the *context model* (which is an abstract representation of the virtual world where the NPCs are placed), (2) the set of *NPCs* involved in the narrative, and (3) the set of basic *capabilities* performable by NPCs to alter the virtual world. For the narrative example of section 2, the context model would be an abstract representation of a room with a door, the NPCs involved are Jerry and George, and the basic capabilities considered are `walk-to-door` and `cross-door`.

Regarding the autonomous behaviors of NPCs, they are authored on the basis of the common physical aspects explained above. These authored behaviors are gathered in autonomous behavior *profiles*. Each autonomous behavior profile is the collection of goal-oriented and context-dependent autonomous behaviors that an NPC can perform in isolation, so they are authored considering no other NPCs on the scene (e.g., in the narrative example above, the door availability is not reflected in a behavior profile because it is only relevant in the case that more than a character is present[2]). For instance, two different behaviors can be represented for the goal `Leave-room`, depending on whether the character is already at the door (in which case she only needs to cross the door) or not (in which case she needs to first walk to the door, and then cross the door). At least one behavior needs to be authored for each narrative goal that can be assigned to a character.

Coordination regimes are authored separately from the autonomous behaviors of NPCs, also on the basis of the common physical aspects of the virtual world. Two are the constituent components of a coordination regime: the *interdependencies* and the *operational relationships* among NPCs [20]. On the one hand, interdependencies are the various inter-relationships that need to be considered between the basic capabilities of the NPCs. In the original narrative example of Jerry and George, there is an interdependency between the capability `cross-door` of each NPC, due to the door only accepting one person through at a time. However, it could be the case that the author is interested in NPCs

[2] The door is assumed to be always open.

colliding at execution time so some conflicts might arise among them.[3] Therefore, this potential collision would be another kind of interdependency that could be represented in a different coordination regime. On the other hand, the operational relationships among NPCs model the priority to be given to each NPC while coordinating their behaviors.[4] They are represented using a total order among the NPCs. For instance, `(< Jerry George)` is an operational relationship representing that the behavior of Jerry has less priority than George's if any of them needs to be changed due to the applicable coordination regime (therefore, in the example of both NPCs leaving the room, George will leave first if this operational relationship applies).

Both the autonomous behavior profiles and coordination regimes are authored and compiled in separated libraries, so they can be reused and combined to create varied stories with dramatic value. In the next section we present a narrative engine that takes this knowledge as input in order to generate an interesting narrative where autonomous NPCs interact.

4 Narrative Engine

Figure 2 depicts the proposed decentralized and multi-agent narrative engine, which makes use of the encapsulated authoring knowledge explained above in order to build an interesting story where NPCs interact in a common virtual world. There are two types of agents that collaborate to generate the story: (1) several *actor* agents, one for each NPC in the story, and (2) one *performance director* to lead the narrative generation process.

All the agents in the narrative engine (both the actors and the performance director) have access to the common virtual world where the NPCs are placed. Furthermore, each actor has private knowledge about the NPC she performs and its related autonomous behavior profile. The performance director also has private knowledge, which consists of the NPCs' narrative goals for the story to be unfold and the coordination regime through which their behaviors need to be coordinated. The performance director starts the narrative generation process by sending a narrative goal to each actor. Then, each actor uses a *Behavior Selector* module to select her intended autonomous behavior for her narrative goal from the behavior profile of the NPC she performs, taking into account the current state of the virtual world. Afterwards, each actor sends her intended autonomous behavior back to the performance director. The performance director uses a *Coordination Manager* module to merge all the intended behaviors (one per actor) with the applicable coordination regime, giving a coordinated story where NPCs interact as a result. This coordinated story can be sent to a game engine to unfold it in the virtual world (this is a represented with a dashed arrow in the figure).

[3] These potential collisions should be managed by monitoring the execution of the unfolding story, which is something out of the scope of this paper.

[4] The operational relationships explained here are a particular case of those in [20].

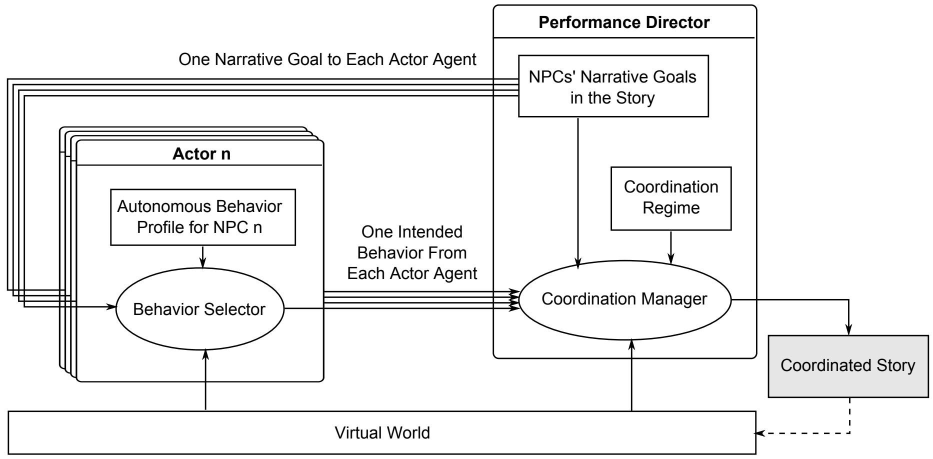

Fig. 2. Decentralized Narrative Engine.

This narrative engine allows for story variability by simply changing one of the encapsulated authoring elements. For instance, three different stories can be generated based on the example of Jerry and George presented in section 2, by simply changing the applicable coordination regime. The first story would be the original example in [20], where George leaves the room before Jerry. In this case, we can use a coordination regime that (1) manages the door availability with an interdependency, and (2) includes the operational relationship `(< Jerry George)`. The second story, consisting on Jerry leaving the room before George can be obtained by simply changing the operational relationship to `(< George Jerry)`. Finally, a third story where both NPCs might collide can be obtained by changing the interdependency so the door availability is not checked. Furthermore, using a cooperative MAP approach allows to distribute the selection of the behavior profiles between the actor agents.

In the next section we present an implementation of the proposed approach (both the authoring framework and the narrative engine) using Hierarchical Task Networks (HTN) planning.

5 Implementation using HTN Planning

Our implementation of the authoring framework and the narrative engine explained above relies in an Hierarchical Task Network (HTN) planning system that has already been successfully used in complex application domains [7, 6, 18] and is commercialized by IActive Intelligent Solutions [9], which is a spin-off from the University of Granada co-founded by one of the authors of this paper. The language used by this planner is based in the Hierarchical Planning Description Language (HPDL) [4], which is an hierarchical and temporal extension of the well-known Planning Domain Definition Language (PDDL) [8]. Furthermore, HTN planning has already been used to successfully represent NPC behaviors [5, 15].

In general terms, an HTN planner takes two inputs: a *planning domain* and a *planning problem*. On the one hand, the *planning domain* encodes the context model (the set of domain objects and predicates that model the world) and the expert knowledge to be considered for generating a plan, which is modeled as a hierarchy of compound and primitive tasks at different levels of abstraction. A primitive task (or planning operator), which is defined by a list of parameters, conditions and effects, is a non-decomposable activity whose execution produces changes in the evolving state of the world. Furthermore, HPDL allows to annotate primitive tasks with metainformation such as a descriptive text. A compound task (or goal) can be decomposed into (compound/primitive) subtasks, following the order constraints imposed by different (and possibly alternative) decomposition methods. A decomposition method, which represents a context-dependent task reduction schema, is defined by the set of conditions that must hold in the current state of the world in order for the method to be applicable. On the other hand, the *planning problem* is composed by the initial state (the state of the world at the beginning of the planning process) and a set of high-level goals (compound tasks) to be achieved. The planning goal is the starting point for the planning process, which explores the space of possible decomposition methods specified in the planning domain, replacing each compound task by its subtasks until the high-level goals are transformed into a partially-ordered set of only primitive tasks that makes up the plan.

With respect to the proposed authoring framework, figure 3 shows the HPDL representation of the authoring elements detailed in section 3 for the narrative example of Jerry and George. The context model is represented with the object types and predicates that model the virtual world where the story unfolds. One of the object types (i.e., `NPC`) is used to represent the NPCs and a *constant* of this type is used to represent each NPC (e.g. `Jerry`). A planning *operator* represents each of the capabilities that an NPC may have (e.g., `walk-to-door` and `cross-door`). These operators are annotated with a narrative text that describe the actions of the NPCs. On the other hand, the autonomous behaviors of each NPC for a given goal are represented using the HTNs formed by *compound tasks* and *decomposition methods* (e.g., an NPC has two different behaviors for leaving a room, depending on whether she is already at the door or not). Therefore, an HTN planning domain (which can be built by joining the context model, set of NPCs, basic capabilities and a collection of autonomous behaviors) is used to represent an *autonomous behavior profile* (see figure 1). Regarding the interdependencies of a coordination regime, they can be represented by *specializing* the planning operators. For instance, the operator `cross-door-interdependency`, which avoids more than one person crossing the door simultaneously, can be included as part of a coordination regime to consider such interdependency. Therefore, we notice that different versions of planning operators can be used for autonomous behaviors and coordination regimes. Thus, the door availability is not reflected in the operator `cross-door`, which is intended to be used within the autonomous behaviors of NPCs. The reason is that requiring the door to be clear in the autonomous behavior of an NPC would reduce the number of interesting narratives that can be generated (for instance, two or more NPCs colliding while trying to cross the door at the same time). Instead, the door availability, which implies an interdependency among NPCs, is represented in an specialized operator to be included in a coordination regime.

```
;;The context model is represented with the
;;objects types and predicates of the domain
(:types NPC room door - object)

(:predicates
  (NPC-in-room ?c - NPC ?r - room)
  (NPC-at-door ?c - NPC ?d - door)
  (door-in-room ?d - door ?r - room)
  (clear ?d - door))

;;Each NPC is assigned a constant
(:constants jerry george - NPC)

;;Autonomous behaviors are represented
;;using HTNs.
(:task Leave-room
 :parameters (?c - NPC)
 (:method isNotAtDoor
  :precondition (and (NPC-in-room ?c ?r)
                (not (NPC-at-door ?c ?d)))
  :tasks((walk-to-door ?c ?r ?d)
         (cross-door ?c ?r ?d)))
 (:method isAtDoor
  :precondition (and (NPC-in-room ?c ?r)
                     (NPC-at-door ?c ?d))
  :tasks((cross-door ?c ?r ?d)))
```

```
;;Each capability is represented with
;;a planning operator.
(:durative-action walk-to-door
 :parameters (?c - NPC ?r - room ?d - door)
 :meta (
  (:tag prettyprint
   "?c walks to the door ?d in room ?r."))
 :duration 1
 :precondition (and (NPC-in-room ?c ?r)
                    (door-in-room ?c ?r))
 :effect (and (NPC-at-door ?c ?d)))

(:durative-action cross-door
 :parameters (?c - NPC ?r - room ?d - door)
 :meta (
  (:tag prettyprint
   "?c leaves room ?r through door ?d."))
 :duration 1
 :precondition (and (NPC-in-room ?c ?r)
                    (NPC-at-door ?c ?d))
 :effect (and (not (NPC-in-room ?c ?r))))
```

```
;;Interdependencies are represented by specializing the basic capabilities of NPCs
(:durative-action cross-door-interdependency
 :parameters (?c - NPC ?r - room ?d - door)
 :meta ((:tag prettyprint "?c leaves room ?r through door ?d."))
 :duration 1
 :precondition (and (NPC-in-room ?c ?r) (NPC-at-door ?c ?d) (overall (clear ?d)))
 :effect (and (not (NPC-in-room ?c ?r)) (at start (not (clear door))) (at end (clear door))))
```

Fig. 3. Example of authoring knowledge represented in HPDL.

With regard to the narrative engine, the state of the common virtual world where NPCs are placed is represented using the initial state of a planning problem (e.g., with predicates like `(NPC-in-room jerry r1)`, `(door-in-room d1 r1)`, etc.), and the narrative goals of each character are represented using planning goals (e.g., `(Leave-room jerry)`). The behavior selector module of the actor agents is a modified HTN planner that uses the planning goal sent by the performance director to decompose the methods in the HTN planning domain that represents the behavior profile of its corresponding NPC until a solution plan is found. The behavior selector keeps track of the decomposed methods and builds the intended autonomous behavior with the abstract HTNs involved in the planning episode. Therefore, each intended behavior is formed by a set of abstract

HTNs. On the other hand, the coordination manager labels these HTNs with the actor where they come from and dynamically merges all of them into a single global HTN planning domain. This merging process links the planning operators that represent the basic capabilities of the NPCs to the specialized capabilities in the coordination regime (instead of using the ones coming from the intended autonomous behaviors of each actor). Then, the coordination manager runs another modified HTN planner (different from the behavior selector) over the goals of all the NPCs by decomposing the methods in this global domain. This modified planner drives its search using the operational relationships of the coordination regime, giving priority to the HTNs labeled with the actors that perform the NPCs with a greater authority. Therefore, the output is a coordinated story that takes the form of a causal and temporal plan where the autonomous behaviors of the NPCs are coordinated according to the applicable coordination regime (e.g., the plan `([(walk-to-door jerry r1 d1)(walk-to-door george r1 d1)](cross-door jerry r1 d1)(cross-door george r1 d1))`, where `()` and `[]` denote sequential and parallel order respectively, can be translated into a narrative where (1) Jerry and George simultaneously walk to the door, (2) Jerry leaves the room by crossing the door, (3) George leaves after Jerry). The use of HTN planning by the behavior selector and the coordination manager ensures the compliance of the story with the constraints imposed by (1) the behavior profiles of the involved NPCs, and (2) the applicable coordination regime.

6 Conclusion and Future Work

We have presented initial steps toward the separation of the autonomous behaviors of NPCs and the coordination regimes that drive their interactions. An authoring framework is proposed and a narrative engine is presented, having both been implemented using a formal planning language and a MAP approach which relies in an HTN planning system. The approach seems promising to reduce the authoring efforts and to provide story variability in IS systems.

The main points for future work are: (1) to author more complex behaviors for NPCs (e.g. as in [15], where NPCs are modeled as instances of professions such as thief, hunter, and woodcutter from which they inherit different behavior profiles); and (2) to augment the coordination regimes (e.g. by considering the interdependencies and operational relationships described in [20]).

Acknowledgments. This work is supported by the Spanish MICINN project TIN2011-27652-C03-03.

References

1. Aylett, R., Louchart, S.: Towards a narrative theory of virtual reality. Virtual Reality 7(1), 2–9 (2003)
2. Aylett, R., Paiva, A., Dias, J., Hall, L., Woods, S.: Affective agents for education against bullying. In: Affective Information Processing, pp. 75–90. Springer (2009)

3. Castelfranchi, C., Miceli, M., Cesta, A.: Dependence relations among autonomous agents. Decentralized AI 3, 215–227 (1992)
4. Castillo, L., Fdez-Olivares, J., García-Pérez, O., Palao, F.: Efficiently handling temporal knowledge in an htn planner. In: International Conference on Automated Planning and Scheduling (ICAPS). pp. 63–72 (2006)
5. Cavazza, M., Charles, F., Mead, S.J.: Character-based interactive storytelling. Intelligent Systems, IEEE 17(4), 17–24 (2002)
6. Fdez-Olivares, J., Castillo, L., Cózar, J.A., García Pérez, O.: Supporting clinical processes and decisions by hierarchical planning and scheduling. Computational Intelligence 27, 103–122 (2011)
7. Fdez-Olivares, J., Castillo, L., Garcıa-Pérez, O., Palao, F.: Bringing users and planning technology together. experiences in siadex. In: Proc ICAPS. pp. 11–20 (2006)
8. Fox, M., Long, D.: An extension to pddl for expressing temporal planning domains. Journal of Artificial Intelligence Research 20, 61 – 124 (2003)
9. IActive: Iactive intelligent technologies. http://www.iactiveit.com/ (2014), accesed 10 February 2014
10. Kapadia, M., Singh, S., Reinman, G., Faloutsos, P.: A behavior-authoring framework for multiactor simulations. Computer Graphics and Applications, IEEE 31(6), 45–55 (2011)
11. Marsella, S.C., Pynadath, D.V., Read, S.J.: Psychsim: Agent-based modeling of social interactions and influence. In: Proceedings of the international conference on cognitive modeling. vol. 36, pp. 243–248. Citeseer (2004)
12. Mateas, M., Stern, A.: A behavior language for story-based believable agents. IEEE Intelligent Systems 17(4), 39–47 (2002)
13. McCoy, J., Treanor, M., Samuel, B., Tearse, B., Mateas, M., Wardrip-Fruin, N.: Comme il faut 2: a fully realized model for socially-oriented gameplay. In: Proceedings of the Intelligent Narrative Technologies III Workshop. p. 10. ACM (2010)
14. Osborne, F.: A new approach to social behavior simulation: the mask model. In: Interactive Storytelling, pp. 97–108. Springer (2011)
15. Paul, R., Charles, D., McNeill, M., McSherry, D.: Mist: an interactive storytelling system with variable character behavior. In: Interactive Storytelling, pp. 4–15. Springer (2010)
16. Porteous, J., Cavazza, M., Charles, F.: Applying planning to interactive storytelling: Narrative control using state constraints. ACM Transactions on Intelligent Systems and Technology (TIST) 1(2), 10 (2010)
17. Riedl, M., Thue, D., Bulitko, V.: Game ai as storytelling. In: Artificial Intelligence for Computer Games, pp. 125–150. Springer (2011)
18. Sánchez-Garzón, I., Fdez-Olivares, J., Onaindía, E., Milla, G., Jordán, J., Castejón, P.: A multi-agent planning approach for the generation of personalized treatment plans of comorbid patients. In: Artificial Intelligence in Medicine, pp. 23–27. Springer (2013)
19. Shapiro, D., McCoy, J., Grow, A., Samuel, B., Stern, A., Swanson, R., Treanor, M., Mateas, M.: Creating playable social experiences through whole-body interaction with virtual characters. In: Ninth Artificial Intelligence and Interactive Digital Entertainment Conference (2013)
20. Smith, B.L., Tamma, V., Wooldridge, M.: An ontology for coordination. Applied Artificial Intelligence 25(3), 235–265 (2011)
21. Von Martial, F.: Interactions among autonomous planning agents. Decentralized AI pp. 105–119 (1990)