Context Management and Intention Mining for Adaptive Systems in Mobile Environments: from Business Process Management to Video Games?

Rébecca Deneckère[1], Charlotte Hug[1], Ali Jaffal[1], Manuele Kirsch Pinheiro[1], Bénédicte Le Grand[1], Raúl Mazo[1], and Irina Rychkova[1]

[1] Université Paris 1 Panthéon-Sorbonne, Centre de Recherche en Informatique
90, rue de Tolbiac, 75013 Paris, France
{firstname.name}@univ-paris1.fr

Abstract. This article describes some recent advances in the Information Systems research domain to support users of mobiles devices such as tablets or smartphones. Indeed, we show how information about the evolving context of users can be organized and analyzed through context management. We also describe an original approach called intention mining, which analyzes traces of users' actions on the system and extracts the rational behind these actions (i.e., the user goals/intentions). Finally, we explain how adequate process modeling formalisms allow for personalized recommendations (and therefore adaptive scenarios) in order to optimize users experience with the information system.
So far, these approaches have mostly been applied to business process models, and we discuss their interest for video games, in particular urban games which combine physical and real worlds.

Keywords: context management, digital traces analysis, intention mining, adaptive processes.

1 Introduction

This article describes some recent advances in the Information Systems research domain to support users of mobiles devices such as tablets or smartphones. Our goal is to provide these users with a better guidance through their daily use of these interfaces. The objective is to generate recommendations that are adapted to a given user's context and that take into account his/her objectives. These recommendations systems also take into account the experience of the other system users.
In order to make this possible, we show how information about the evolving context of users can be organized and analyzed through ***context management***. We also describe an original approach called ***intention mining***, which analyzes traces of users' actions on the system and extracts their goals (intentions). Finally, we explain how an adequate ***process modeling formalism*** allows for personalized recommendations (and

therefore adaptive scenarios) in order to optimize users experience with the information system.

So far, these approaches have mostly been applied to business process models, and the second objective of this work is to study their interest for video games, in particular urban games which combine physical and real worlds.

2 Context Management

2.1 Context management in information systems: state of the art

The "one size fits all" way of using Information Systems (ISs) does not work anymore. Users now expect these systems to be context-aware, i.e., to adapt to users' specific contexts [Baldauf et al., 2007]. The context represents any information that can be used to characterize the situation of an entity (a person, place, or object) that is considered relevant to the interaction between a user and an application [Dey, 2001]. More generally, we can consider the context as everything that surrounds the center of interest of the individual and that provides additional information that can help understand his/her focus [Mostefaoui et al., 2004].

Gamers playing to an urban game are very sensitive to their contexts. When playing their game, they evolve outdoors in the real word and, as a result, can modify their behavior to match this context. For instance, if the gamer is heading to a particular place but that rain starts pouring, then he will certainly delay his walk to do something else until the rain stops (exchange with another gamer, try to solve a clue, or any other activity that can be done inside instead of outside a building).

The management of context information has become strategic in the context of pervasive information systems [Kourouthanassis et al., 2008] [Najar et al., 2009] [Villalonga et al., 2010]. Data generated from these systems by observing user interactions and the environment in which they take place is extremely valuable for systems designers and service providers. It serves as a basis to enhance their systems and services according to user's context (e.g., location, date, time, past and current activities). In the case of urban games, context can be composed of the geolocation of the gamer, the time of day (or night), the proximity of other gamers, the weather forecast, etc.).

Analyzing the impact of context information on the user's actions becomes a key aspect for successful pervasive information systems. Data mining techniques can be used to this end, in order to extract relevant information to support decision and prediction [Fayyad et al., 1996] [Chen et al., 1996]. Several works such as [Ramakrishnan et al., 2014] are considering data mining techniques for analyzing context data. These techniques can play an important role in understanding and discovering the intrinsic relationships between data. Among them, Formal Concept Analysis (FCA)

[Priss, 2006] [Wille, 2005] represents an interesting method of conceptual clustering. It helps extracting implicit knowledge and finding a natural data structure, while associating user actions to observed context elements. We describe in the following section how FCA can be used to manage context information.

2.2 Use of Formal Concept Analysis for Context Management

The ways mobile interfaces are used depend a lot on user context: geographical location, time of the day, day of the weeks, meteorological conditions, quality of network connection, etc. understanding the influence of these context elements makes it possible to propose better personalization and recommendation features and to improve adaptation mechanisms. However, identifying relevant context elements (for a given user) from the highly heterogeneous collected context data is a challenge not only because of data heterogeneity, but also because this relevance is personal and may vary significantly from user to user. Context analysis methods are then necessary in order to establish context relevance for a given user. In earlier work [Jaffal et al., 2014], we have used Formal Concept Analysis (FCA) to this end and shown its interest to cluster context elements with regard to associated user activities (and vice versa) into overlapping classes.

Table 1. Example Formal Context (the values in parentheses correspond to the number of times the application was used in each context).

Applications	University	Restaurant	Parents' home	Home	Transportation	3G	Morning	Coffee break	Lunch break	Afternoon	Evening
Gmail	1 (2)	0	0	0	1	1 (2)	1	0	0	1	0
SMS	1 (80)	1	1 (5)	1 (30)	50	1 (170)	1 (40)	1 (20)	1 (40)	1 (60)	1 (10)
Tele-phone	0	0	0	1 (3)	1 (2)	1 (5)	1	0	0	1 (2)	1
VDM	0	0	0	1 (4)	1 (2)	1 (5)	1 (4)	0	0	0	1 (2)
Flappy Bird	1	0	0	1	1	1	1	1	0	1	1
Youtube	0	0	0	1	0	1	1	0	0	0	0

The input of the FCA algorithms is a binary relation - called *formal context* - between a set of *objects* and a set of *attributes* that describes these *objects*. Table 1 shows such a formal context in which objects are the applications executed by a user on a tablet,

and attributes describe the context in which these applications have been used[1]. For example, *Youtube* has been used at *home*, but also on a *3G* network and in the *morning*. Note that the only possible values in the formal context are 0 and 1, which does not reflect the real numbers of time various applications have been used (indicated in parentheses): *SMS* are associated to *university* and *restaurant* in a similar way in the formal context, although the number of *SMS* sent in each location is very different.

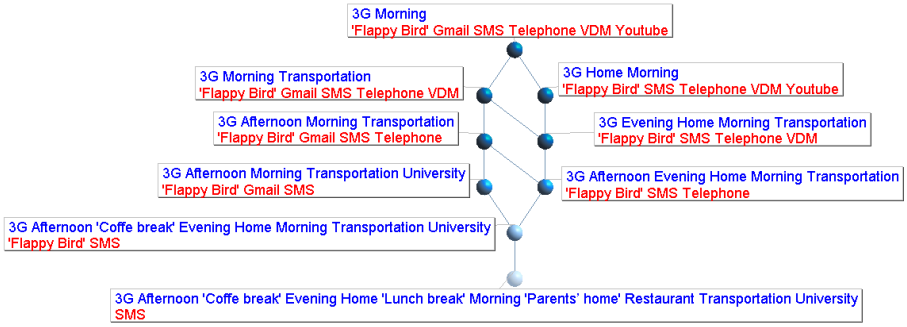

Fig. 1. Galois lattice generated from the formal context of Table 1

Figure 1 represents the Galois lattice generated from the formal context of Table 1. Each cluster of the lattice is called a (formal) concept which groups objects that have attributes in common. For example, this lattice shows that *Flappy Bird, SMS* and *telephone* applications have all been used in the *morning*, in the *afternoon*, in the *evening*, during *transport*, from a *3G* network and at *home*. These context elements are common to these applications. One interest in terms of recommendation is that if the user happens to be in a similar context in the future, the information system can automatically personalize the interface according to the user preferences, e.g., display a game in the foreground of the screen when the user enters the metro station.

Note that the concepts become more and more specific when going from the top to the bottom of the lattice, from concepts with many applications sharing few context elements to concepts with few applications sharing many context elements. Indeed, the concept containing *Flappy Bird, SMS* and *telephone* applications can be specialized into a concept that only contains *Flappy Bird* and *SMS* applications, with two additional context elements: *coffee break* and *university*.

The lattice provides a clustering of applications according to the context elements they have been associated to. Conversely, context elements are described with regard

[1] This data comes from a dataset built from a questionnaire filled by 28 master students of our university. The goal of this survey was to know in which context they had been using applications (such as social networks, games, emails, etc.) on their tablets during a week.

to the applications associated to them. We can notice from the lattice that *3G* connection and *morning* context elements appear in all concepts, which means that they are not significant here and should not be taken into account for recommendations to this specific user. Conversely, the *SMS* application has been used with all context elements, and is therefore context-independent.

We have proposed a methodology to reflect in FCA results the relative importance of context elements on user activities, by refining the input data using a semantic frequency measure.

In order to build a formal context from collected data, Formal Concept Analysis requires to replace the numeric values (Table 1) by binary values in order to represent the relationships between the objects and the formal context attributes in concept lattice. In table 1, we have followed a simplistic approach, by directly replacing null values indicated by users by a zero and any positive value by one. More elaborate methods exist: Pensa et al. [Pensa et al., 2004] use a cut-off threshold to replace values greater than or equal to the threshold values by 1 and the rest by 0. The method in [Ma et al., 2006] is to set a threshold δ, and then turn the context into a binary one by replacing values less than δ with 0 and 1 the others. However, these thresholds are calculated based on numerical values in the formal context regardless of the semantic link between the attributes.

The originality of our frequency measure described in [Jaffal et al., 2015] is that it is a semantic measure that relies on a context ontology, illustrated in Figure 2.

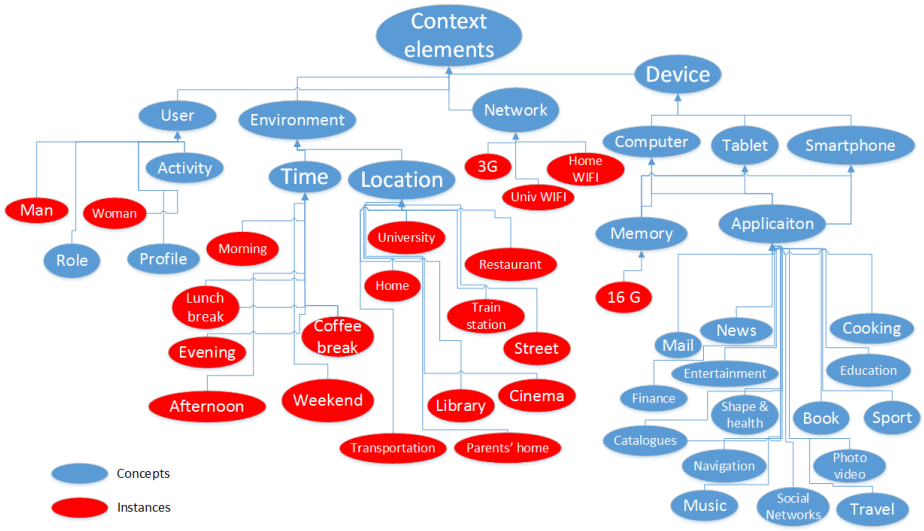

Fig. 2. Context Ontology

The lattice represented on Figure 3 is obtained after the refinement process we propose. It is more interesting than the original lattice of Figure 1 for recommendation and personalization purposes, as more clusters of applications and context elements are generated, leading to a finer-grained classification. In Figure 1 for example, *Flappy bird, SMS, telephone, VDM* and *Youtube* applications have in common the context elements *3G, home* and *morning*, whereas they are separated into 2 concepts in Figure 3: *Gmail, VDM* and *Youtube* are associated to *3G* and *morning*, and F*lappy bird, telephone, VDM* and *Youtube* are associated to *3G* and *home*. It means that after the refinement strategy, *Flappy bird* and *telephone* are no longer associated to the *morning* (and will therefore not be considered as very likely to be used in the morning). We also notice that some context elements have disappeared from the lattice, as they have no sufficient impact on the user applications: *coffee break*, *restaurant*, *parents' home* and *lunch break*. It is interesting to remark that the number of concepts in the lattice increases despite the disappearance of 4 context elements. This shows that the precision of the obtained results has indeed increased significantly with regard to the initial lattice.

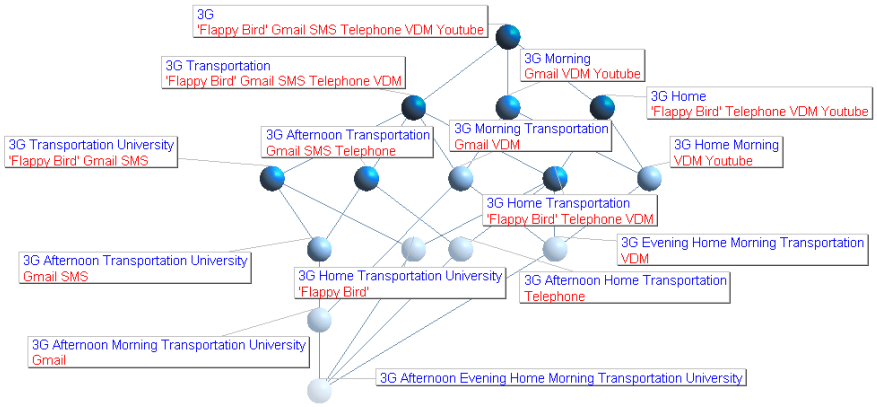

Fig. 3. Galois lattice obtained after the semantic refinement strategy of the formal context

2.3 Interest for Video Games

We have shown previously how the semantic refinement strategy we proposed leads to a better understanding of context impact. Such understanding represents an important step towards personalization and recommendation features. Urban games have many common features with pervasive information systems: users interact with their system in a transparent and mobile way. Examples of context elements that could be relevant in urban games are shown on Figure 4.

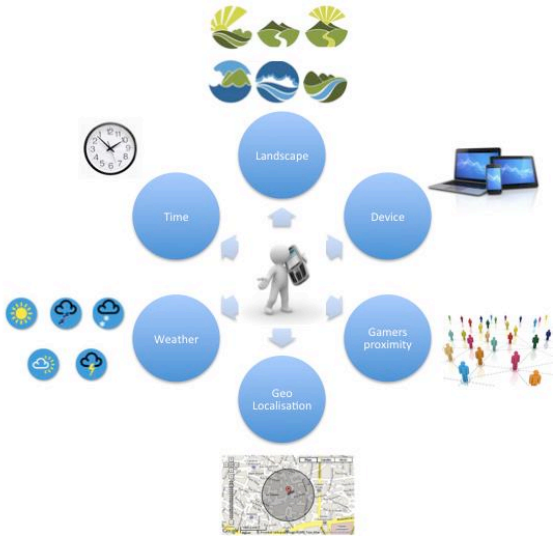

Fig. 4. Example of context elements in urban games

The application of the approach we have proposed in the previous section is illustrated on Figure 5.

Fig. 5. Context management applied to urban games

This methodology could be could be interesting in order to make the following actions possible:

1) Analysing gamers' behaviour in similar contexts (Figure 6)
For instance, we can imagine that when gamers are located in a specific place and use a specific device, they systematically choose to perform a precise activity, whereas if

their context changes (for instance same place but other device), they choose to do something else. Information about gamers' behavior according to their context can be very useful for game designers in order to propose different kinds of services accordingly. We can imagine that the expected behavior is the one obtained with the use of a smartphone, and propose a new clue to tablet users to help them perform the expected activity.

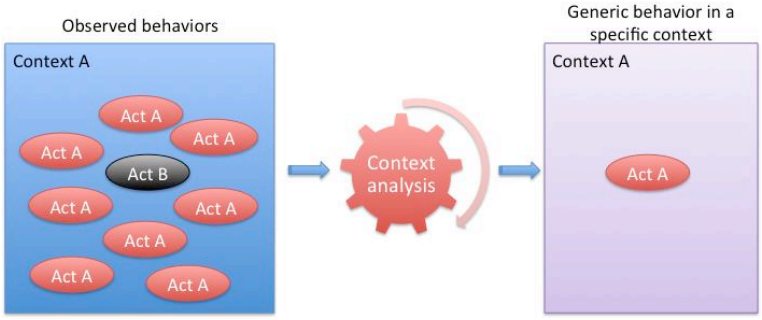

Fig. 6. Analysing gamers' behaviour in similar contexts

2) Analysing the contexts generating a similar behavior (Figure 7)

In a similar way, game designers could be very interested in knowing the contexts leading to the same behavior. Let us consider a key activity in a game, which a set of players has a lot of difficulty to perform. The analysis of the players' contexts may highlight combinations of context elements that confuse these gamers and prevent them from completing the task correctly. As with the previous example, the game designer could then decide to offer a clue to the players who are within these contexts to help them go further in the game.

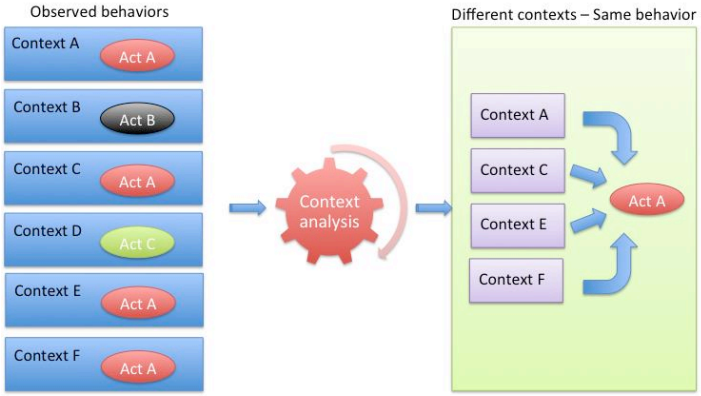

Fig. 7. Analyzing the contexts generating a similar behavior

3) Recommending specific actions according to the gamer's context (Figure 8)

Context analysis, performed at run time, can also be a way to recommend some activities to the gamers, according to their current context. For instance, taking into account weather modifications, the gamer can be proposed in-door or out-door quests, to keep his interest in the game by playing in comfortable conditions.

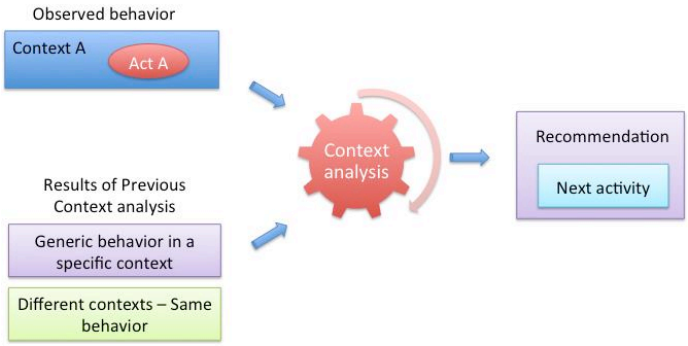

Fig. 8. Recommending specific actions according to the gamer context

3 Digital Traces Analysis and Intention Mining

Although context management is essential to understand the interactions of users with an information system (and, similarly, the actions performed by players of video games), we believe that it is also necessary to capture the intentions of the users when they perform actions. Indeed, we claim that the actions executed by information system users follow a strategy that aims at satisfying an intention within a specific context. Context management has been addressed in the previous section; we know focus on the identification of users intention through a process we call *intention mining*, described in Section 3.1.

3.1 Process and intention mining: state of the art

Analyzing activities in the Information Technologies (IT) context is a very lucrative business. While surfing on websites, all our activities—products and services searches, on-line game activities, comments and posts on blogs or social networks, among others—are traced and analyzed to propose us the most adapted advertisements.

Games, as other kind of software, leave digital traces of the users that can be observed, stored and analyzed, as illustrated on Figure 9. Video and computer gaming have become increasingly popular branches of the entertainment industry and lots of traces are generated. Due to the large variety of available games, players have a plethora of choices. User selection can be influenced by the content of games, the quality of the services, the popularity of games, the choices of in-game friends, etc. Retaining players with longer player lifetime can yield more revenue for game com-

panies. The ways a player interacts with in-game friends, as well as his/her in-game play strategy are also some interesting facts that can be analyzed in the traces.

Tracing the gaming activities of a user can then become the source of a lot of knowledge. Analyzing game traces allows understanding the characteristics of games such as player behaviour [Suznjevix et al, 2011][(Ducheneaut et al, 2007], traffic analysis [Chen et al, 2006][Kinicki et al 2008], resource management [Chambers et al, 2010][Denault et al, 2011], etc. Traces can be stored in a specific repository as the Game Trace Archive[2] which allows analyzing large and public game traces datasets [Guo et al, 2012].

Fig. 9. Digital traces capture

The player behavior, as any user behavior, can also be described as a process, and process can be defined with process models. The approach of tracing and analyzing activities has been used with great success in the field of process mining, in particular for enhancing software development. [Weijters and van der Aalst, 2003] have proposed methods and tools based on Petri nets and α type algorithm to analyze events logs (traces) in order to discover process models. Process mining approaches aim at modelling users' behaviors in terms of sequences of tasks and branching in an automatic way [Van der Aalst et al., 2004] [Agrawal et al., 1998] [Cook and Wolf, 1998] [Datta, 1998] [van Dongen, 2004] [Herbst, 2000]. The resulting (i.e., mined) process models are activity-oriented. However, activity-oriented models lack flexibility[3].

Processes can also be seen as teleological [Ralph and Wand 2008]. A teleological process is a process that takes into account the teleological behaviors of process enactment (behavior attached to the notion of goal). It describes the intentions (goals, objectives) associated with a result that an individual wants to obtain. Many research works in intentional process modelling demonstrate that the fundamental nature of processes is mostly intentional and the processes should be modelled from an intentional point of view [Davis et al., 1989] [Plihon, 1996] [Rolland et al., 1999]. The notion of context also plays a key role for the intention, since a given intention

[2] http://gta.st.ewi.tudelft.nl
[3] We will further discuss process models in Section 4.

emerges in a given context, which not only promotes its appearance, but also influences the realization of this intention [Rolland, 2005]. Process mining applied to intentional process models is called *intention mining*, which is at the heart of our solution. This original concept of "Intention Mining" aims at automatically discovering intentions from traces of user activities [Hug et al., 2012] using pattern mining [Zaki, 2001]. Hidden Markov models technique [Baum et al., 1966], a specific type of Dynamic Bayesian Networks, was used in the Map Miner Method [Khodabandelou et al., 2014] [Khodabandelou, 2014]. Moreover, [Epure et al., 2014] proposed an approach using an unsupervised technique and a knowledge tree to build intentional process models at a low level of abstraction. Information about user intentions within a specific context, together with knowledge of other user behaviors with the same intentions in similar contexts is the key to provide them with adapted recommendations.

3.2 Potential interest of intention mining in the area of video games

In this section, we describe how the intention mining techniques presented previously could be applied in the context of video games. Player behaviors traces, analyzed with intention mining techniques, could in particular make the following activities possible:

1) Discovery of the underlying player's intentions and strategies hidden in the traces.

Intention mining allows identifying whether there exists a *generic* behavior of the players, through the identification of players' usual activities and the usual ordering of these activities, as shown in Figure 10. For instance, if gamers usually change their direction after finding a clue, this information could be useful for the game designer in order to offer new services adapted to this generic behavior (for instance to automatically display a map when the user enters the clue in its device). Moreover, from a marketing point of view, knowing the gamer's general behavior can add some highlighting information when designing ads for the game.

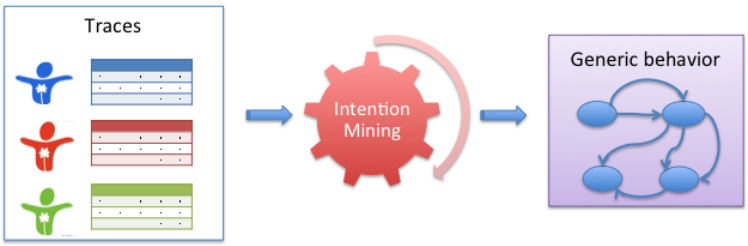

Fig. 10. Extraction of a general player behavior

2) Conformance checking: comparison between the observed behaviors of players and what is expected from them by the game designers.

Once the actual behavior of players is identified, it is possible to compare it with the one expected by the game designer, as shown on Figure 11. Some unexpected issues may appear, such as a gamer intention never achieved, a strategy never used, a new intention that was not expected by the game designer, etc.

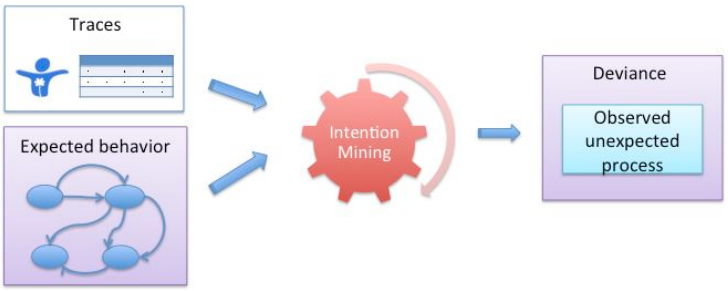

Fig. 11. Comparison between actual and expected players behaviors

3) Enhancement of the gaming process model

If the observed player's behavior is not consistent with the game designer's goal, e.g., if some paths are never followed, the game designer can add features in the game to guide players towards these paths (see Figure 12). Conversely, if some strategies used by the players are more efficient than the ones expected by the game designer, the expected model can be enriched by adding the identified new behaviors in the model.

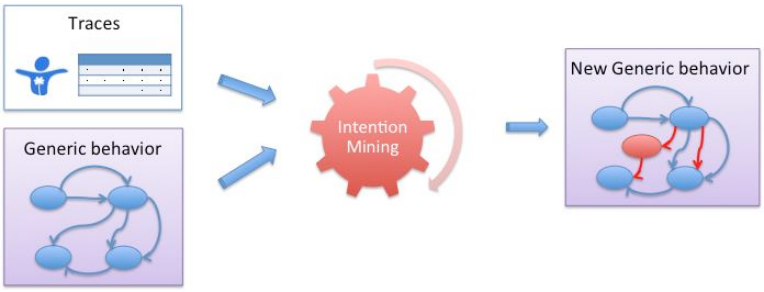

Fig. 12. Refinement of the player process model

4) Recommendation of specific steps to the gamer, at run time, based on their supposed reasoning while performing activities and based on their current context.

It is possible to recommend the next activity to a gamer, following the percentage of players that executed this activity when exhibiting the same behavior earlier in the game (see Figure 13). For instance, if the generic behavior consists in exchanging information with other gamers just after a specific step, the device used by the gamer

can propose this action to him automatically as the next step option. The game can then become more adaptable to the gamer behavior, at run time.

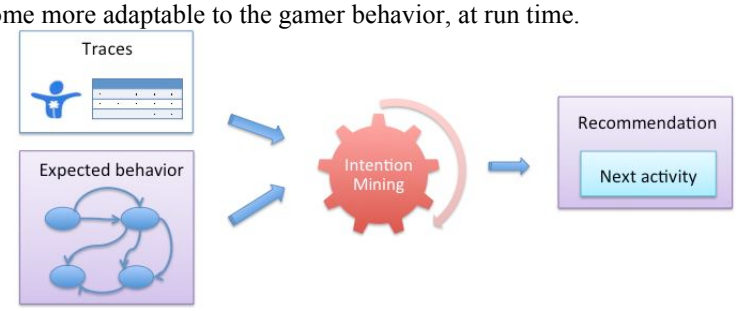

Fig. 13. Recommendation to players based on context and intention

4 State-Oriented Adaptive Process Modeling

Sections 2 and 3 of this paper are dedicated to the analysis of user context and intentions. These elements allow for personalizing user interaction with their mobile interface and for supporting them with relevant recommendations based on their context intentions and previous experience.
We now focus on the adequate way to model the underlying scenarios followed by users during their interaction with their mobile interfaces. We first discuss the challenges of game scenario modeling, then we briefly overview the modeling formalisms used in the domain of software engineering and Business Process Management. Finally, we discuss the interest of statecharts and life sequence charts (LSC) modeling formalisms for the domain of urban games.

4.1 Challenges of game scenario modeling

The objective of a multi-player game is to create a virtual or semi-virtual (in the case of urban games which mix virtual and real worlds) environment where players are placed. During the game, players follow some scenarios in order to achieve one or multiple goals through communicating and collaborating with each other.
A game scenario needs to specify the objects, their positions in a virtual world (e.g., a maze) and some puzzle to be solved or some task to be carried out. It induces a partially ordered set of actions that the player must perform [Gal et al., 2002]. In general, the game scenario is a description of the main game phases and the navigation between them.

The main challenge in game scenario is to remain "unpredictable" for the player while driving this player towards some (probably implicit) goal. Unpredictability is important: first of all, if similar situations occur too often and if the reaction of the game

environment and its objects can be predicted, the player may get bored very rapidly. The scenario of the game needs to evolve together with the player.

Unpredictability is extremely important in urban games which mix real and virtual worlds, as the artificial life needs to bear a resemblance to real life. As real life is rich, unpredictable and evolving, the player needs to feel it in the simulated game environment too. Not only the situations encountered by the player should look "unique" but also the opportunities (what the player can and cannot do) in each particular situation should be defined dynamically and not embedded into the scenario. The game scenario therefore needs to unfold progressively, at run-time, and all players can be also considered as scenario designers.

The game scenario thus can be compared to a process, for which the requirements for flexibility and "unpredictability" support are also relevant. Game designers can therefore reuse process modeling languages to model the game scenarios. In the following section, we review existing modeling paradigms used for business process management.

4.2 Process modeling: state of the art

Conventional activity-oriented formalisms for process modeling represent a process as a sequence of activities. These formalisms have been mentioned in Section 3. The most popular examples of these formalisms include BPMN and UML activity diagrams. These formalisms are mostly prescriptive and provide only limited support for process adaptability at run-time.

Goal-oriented is another group of modeling formalisms that, compared to activity-oriented formalisms, represent a process as a sequence of goals. The MAP modeling notation mentioned earlier in this paper is an example of goal-oriented languages. Context-driven goal-oriented process models such as [Rolland et al., 1995], [Soffer and Yehezkel, 2011] and [Pohl and Weidenhaupt, 1997] support automated recommendations and user guidance, providing that for each goal all the situations (states) in which this goal is achievable are known.

In urban games, we might face unpredictable sequences of events and non-repeatable execution scenarios; it would therefore be hard, if at all possible to model relations between various process situations, goals and activities that must/can be executed in order to achieve these goals.

In the literature, several major perspectives of the process models are specified [Jablonski and Bussler, 1996]: the control flow perspective that captures the temporal ordering of process tasks, events, and decision points; the data perspective that captures the lifecycle of data objects (creation, usage, modification, deletion) within the process; the resource perspective that describes how the process is carried out within

the organization and deals with roles and resource assignments; the operational perspective that addresses the technical aspects of process execution and specifies the elementary process tasks and their assignment to concrete applications or application components of the organizations; the context perspective that describes the attributes related to the process execution context; the performance perspective, addressing the process cost effectiveness.

Flexibility along control flow, data and resource perspectives is widely addressed: modeling languages (e.g., BPMN, EPS, C-iEPC, Declare) enable the specification of processes where the activities can be skipped, repeated, or their ordering can be chosen depending on the context (at run time). Some languages (e.g., configurable models in C-iEPC [La Rosa et al., 2009]) allow for dynamic resource and data management. Flexibility along the context and the operational perspectives, however, are still challenging.

Flexibility along the operational perspective would allow the player, for example, to not only select an activity to execute from the predefined list, but also to propose an alternative that was probably not even thought of at design.

To search for a formalism that would support such flexibility, we appeal to the domain of complex reactive systems.

According to [Harel et al., 2008], the design of games and complex reactive systems share a common set of challenges: 1) The need to construct a system from a set of many interacting small units; 2) The importance of emergent properties, unique characteristics of the global system behavior that emerge from the interaction between the small units but are not always evident from the rules of behavior of each of the components by its own; 3) The crucial role of visualization and graphics; and 4) the need to provide strong tool support that frees the mind and encourages creativity of the designers, allowing them to focus on the big picture and the important parts rather than on implementation details.

In the games where multiple players are placed in virtual reality (i.e., an “artificial life”), each "object" or “actor” of the game is specified as a random automaton sending and receiving stimuli from other objects. The global behavior is the results of the interactions between local ones [Gal et al., 2002]. This allows for simulating complex behaviors and avoiding repetitive scenarios.

For the specification of unpredictable and flexible game scenarios, the state-oriented modeling paradigm and the formalisms of statecharts [Harrel, 1987] can be beneficial.

4.3 Applying the statecharts modeling formalism to game processes

Within the state-oriented paradigm, a process (a game in our case) can be described by a set of states and transitions between these states. A game starts at an initial state and terminates at a final state. A state transition is triggered when some condition is fulfilled. The sequence of states and transitions that leads from the initial state to the final state can be seen as a game scenario.
Activities carried out by a player depend on the current state of the system (the game environment in our case) and the game scenario is adapted at run time, according to the evolution of the game.

Examples of state-oriented modeling formalisms include state machines in UML [Rumbaugh et al., 2004], generic state-transition systems or state machines, such as FSM [Plotkin, 1981] or Petri Nets [Murata, 1989], and statecharts by D. Harel [Harel, 1987] created for the specification and analysis of complex discrete-event systems.

The core difference between state-oriented (and goal-oriented) formalism is the way they specify the transition between states (or goals) within a process: whereas some formalisms (e.g. Petri Net) state that a transition is a result of an execution of an activity, the others (e.g., statecharts) state that a transition is a result of a triggering event. In the first case, all activities need to be explicitly modeled and thus these formalisms do not provide much flexibility compared to activity-oriented formalisms mentioned above. Whereas in the second case, only the (desired or undesired) triggering events have to be specified. The activities that "produce" these events may remain unknown and can be defined at run time. This provides a supplementary degree of flexibility along the operational perspective required by urban games.

Following this paradigm, a game scenario specification can be divided into two parts: the state-transition part, defined with a set of states and transitions between states and their triggering events, and the activity part, defined by a list of activities specified by their preconditions and outcomes. The process enactment can be seen as a dynamic selection of activities to produce some outcomes (events) that make the process progress towards its (desired) final state.
In [Rychkova et al., 2015], we proposed the basis for implementing the state-oriented formalism of statecharts for modeling case management processes, where the case management scenario, similarly to a game scenario, depends a lot on the context and requires flexibility. In [Kushnareva et al., 2015], we elaborated on this basis, proposing an approach for interactive modeling and simulation-based analysis of process specifications for crisis management. Again, crisis handling can be easily considered as a subject of a game. This approach is based on statecharts formalism and YAKINDU statecharts tools[4].

[4] Yakindu Statechart Tools version 2.4. for Eclipse Luna, 2014, http://statecharts.org/

The state-oriented paradigm allows us to exclude activities from the process design: we can state that "any activity is good as soon as it produces a desired outcome". In particular, it enables deferred activity planning, that gives more freedom to the process participant in choosing an activity that is adapted for a concrete situation.
The system based on statecharts model cannot be considered as a “management” system that automatically defines and executes the scenarios, but rather a “recommendation” system that only advises the process participant on scenario planning but not drives him/her through a sequence of predefined steps. That is why this formalism is also beneficial for game design.
In our vision, the ”a priory” statecharts specification of a game can be analyzed using graph theory algorithms. The objective of the analysis is to search and optimize a path from some current state of the statecharts model to its target state, representing the goal of the game. As a result, the ”best next state to visit”, ”best next transition to fire” and, consequently, ”possible activities to execute” are recommended to the player.
The state-oriented model will address the internal behavior of objects involved in the multi-player game in a simulated virtual world. The internal behavior of an object describes how this object can react on any external stimuli in any context (i.e. situation).

This model can be complemented by a model of inter-object behavior [Harel et al., 2008] that describes how this object interacts with other objects in particular situations. In [Harel and Marelly, 2003], the inter-object behavior is modeled with scenarios that specify the interactions between objects.

In [Harel et al., 2008], the authors state that “the approach makes it possible to describe behaviors of different parts of the system using different reactive system design languages and tools. Among the advantages of this approach is the ability to use existing analysis tools to understand the game behavior at design time and run time, the ability to easily modify the behavior, and the use of visual languages to allow various stakeholders to be involved in early stages of building the game.”

Let us illustrate our approach on a simple example inspired from urban games. Figure 14 represents a statechart that specifies a (gamer’s view on a) game with a state machine defined at three hierarchical levels. On the higher level, only two states are defined: *Disconnected* and *Connected*. The transitions between these states are triggered by the events produced by a gamer: *e1* – when the gamer connects to the game, and *e2* – when he/she disconnects.

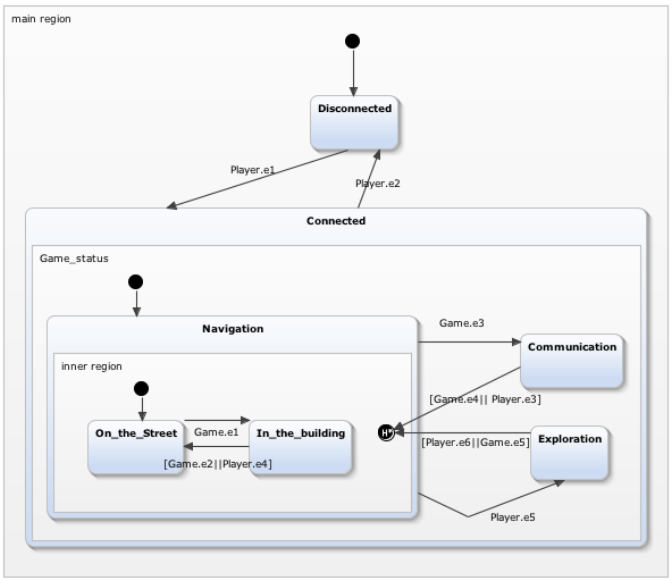

Fig. 14. Example statechart inspired by urban games

When *connected* (the second hierarchical level), the gamer can be in one of the three (exclusive) states: *Navigation*, when the player moves in a physical environment (e.g., walking on the street or visiting buildings, as shown on the third hierarchical level) with an objective to pursue another player, to reach some specific place defined by a mission or to search for a clue; *Communication*, when the player meets an actor or another player with whom he/she can communicate in order to search for a clue; *Exploration*, when the gamer explores the environment around him/her (e.g., a building) in order to search for a clue.

The transitions between these states are defined by logical expressions over events (*e1, e2*, …) . The list of events is presented in Fig. 15. Events can occur as a result of a player's actions (these events are prefixed with *Player* in the diagram) or can be produced by the game environment (these events have the *Game* prefix).

A black circle in the diagram identifies a substate that will be entered "by default" when its parent state is visited: for example, when the *Navigation* state is entered for the first time, its substate "*On_the_street*" is activated.

A black circle with a H* symbol means "entering the state by history": for example, once the gamer leaves the *Communication* or the *Exploration* state, he/she returns to his/her previous position in the Navigation state.

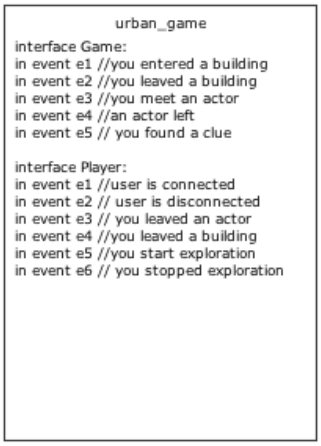

Fig. 15. Events list

The main idea is that the *activities* that the gamer can execute are not specified at all by this model. It only specifies the *events* that can trigger state transitions. The player is free to perform any activities, assuming that they will lead to the generation of a desired event. For example, in the *Navigation* state, the gamer can walk, run, stop, turn, take a metro, ride a bicycle etc. Along these lines, while in the *Exploration* state, the gamer can use any means at his/her disposal in order to search for a clue. This means that concrete activities can be defined by the player based on particular situation. This is especially important for urban games, where the game is taking place in a real environment.

Obviously, this example is very simple. When the scenario becomes more complex, more specific events associated with the game mission will be defined and the player behavior will be specified with more details using new hierarchical levels.

The statechart represented on Figure 14 has modeled with the YAKINDU open source environment for modeling and simulating statecharts. This model can be simulated and refined, providing more details about the game: new hierarchical levels, states, events, and transitions can be specified.

5 Conclusion and Perspectives

We have presented how context management, intention mining and state-oriented adaptive process modeling could be beneficial to game process analysis and modeling, in order to better understand the behavior of players during their gaming experience and provide them with relevant recommendations. This section first summarizes

the contributions presented in this article. Finally, as the recent advances presented in this article have been developed independently, we present one of our perspectives for future work, which consists in integrating them into a single solution.

5.1 Conclusion: overall approach

The overall objective of the various contributions presented in this article is to provide users of an information system with relevant recommendations, which take into account their context and their intentions, as illustrated on Figure 16. We propose to build a recommendation system based on the model of user behavior using a state-oriented modeling formalism that integrates context information and intentions of users. User intentions are extracted from traces with intention mining algorithms, whereas context management relies on Formal Concept Analysis.
The obtained statecharts can then be used in order to identify the next best state for the process according to the current situation of the user and recommend the most efficient path towards this next best state.

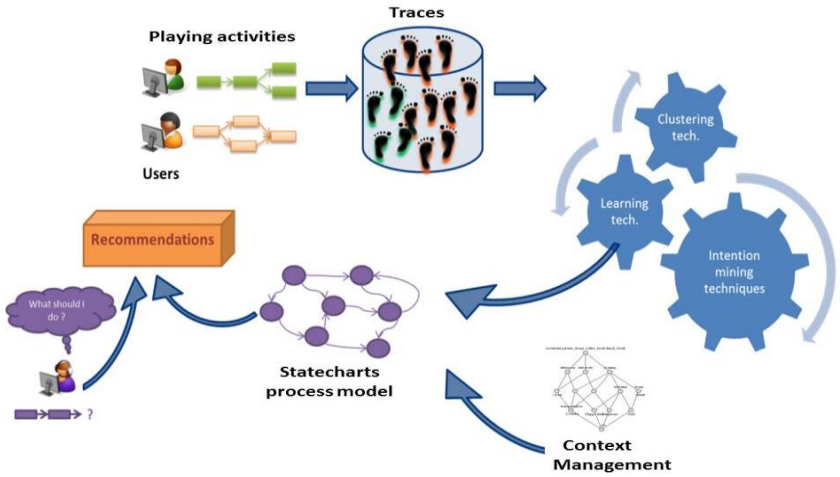

Fig. 16. Overall approach

5.2 Perspectives: towards an integrated solution

To achieve autonomic computing, IBM has suggested a reference model for autonomic control loops [IBM, 2003], which is usually called the MAPE-K (Monitor, Analyze, Plan, Execute, Knowledge) loop, illustrated on Figure 17. The MAPE-K loop is similar to the generic agent model proposed by [Russel and Norvig, 2003], in which an intelligent agent perceives its environment through sensors, and analyzes the obtained information to determine actions to execute on the environment.
In the MAPE-K loop, the adaptive system represents any software or hardware resource that can be adapted to better responds to the changing context. Thus, the adap-

tive system can for example be an operating system, a specific software component in an application (e.g., the query optimizer in a database), a cluster of machines in a grid environment, a wired or wireless network, a middleware, but also an adaptive game.

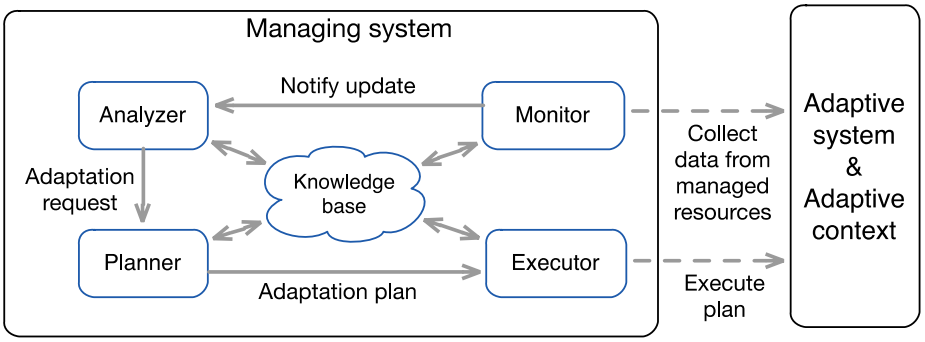

Fig. 17. MAPE-K loop architecture

The elements of the MAPE-K loop and their interactions are presented as follows:

- The ***monitor*** function collects the details from the internal and external managed resources. For instance, the managed resources in an urban game can be metrics (e.g., offered capacities or powers, throughput of the network for distributed games, and score), configuration property settings and so on. In addition, the monitor function periodically aggregates, correlates, filters this information and sends it to the ***analyzer***. With regard to the contributions presented in this article, the monitor is responsible for context management and the monitored resources are those that appear in the context ontology.
- In our case, the ***analyzer*** function performs the intention mining. Moreover, it evaluates whether changes are required in the adaptive system (i.e., whether recommendations are needed in the game) according to gamer intentions and to the information provided by the ***monitor*** function. Usually, changes are required when the current system does not satisfy the constraints provided by the ***monitor*** or when a better solution exists that satisfies these new constraints. In our case, the ***analyzer*** is in charge of finding the next best state in the statecharts process model, relying on context information and on gamer intentions (which are part of the ***knowledge***). If changes are required in the system (i.e., in the game), a change request is sent to the ***planner***.
- The ***planner*** function structures the actions needed to achieve goals and objectives. The planner creates or selects a procedure to enact a desired alteration in the managed resources. The ***planner's*** role is to identify the path that should be taken in order to reach the next best state. Thereafter, the plan is passed to the ***executor***.
- The ***executor*** function changes the structure or the behavior of the adaptive system (the game), and sometimes also of the adaptive context [Sawyer et al., 2012],

according to the information provided by the ***planner***. In our case, it consists in actually providing users with recommendations.

- The ***knowledge*** base contains the data shared by the monitor, analyzer, planner and executor functions. This shared knowledge, is created/updated by the monitor, analyzer and executor. In our case it gathers in particular the context ontology, and the behavior model.

References

1. Agrawal, R., Gunopulos, D., & Leymann, F.: Mining process models from workflow logs (pp. 467-483). Springer Berlin Heidelberg (1998)
2. Baldauf, M., Dustdar, S., Rosenberg, F., A survey on context-aware systems, Int. J. of Ad Hoc and Ubiquitous Computing, 1 2(4): 263-277 (2007).
3. Baum, L. E., Petrie, T. Statistical Inference for Probabilistic Functions of Finite State Markov Chains, The Annals of Mathematical Statistics 37 (6), 554–1563 (1966)
4. Chambers, C., Feng, W.C., Sahu, S., Saha, D. and Brandt, D.: Characterizing online games. TON, 18(3):899– 910 (2010)
5. Chen, K.T., Huang, P., and Lei, C.L.: Game traffic analysis: an MMORPG perspective. Computer Networks, 50(16):3002–3023 (2006)
6. Chen, M.S., Han, J., Yu, P.S., Data mining: an overview from database perspective. IEEE Transactions on Knowledge and data Engineering, 8(6): 866-883 (1996)
7. Cook, J. E., and Wolf, A. L.: Discovering models of software processes from event-based data. ACM Transactions on Software Engineering and Methodology (TOSEM), 7(3), 215-249 (1998)
8. Datta, A. (1998). Automating the discovery of as-is business process models: Probabilistic and algorithmic approaches. Information Systems Research, 9(3), 275-301.
9. Davis, F. D., Bagozzi, R. P., & Warshaw, P. R. (1989). User acceptance of computer technology: a comparison of two theoretical models. Management science, 35(8), 982-1003.
10. Denault, A., Canas, C., Kienzle, J., and Kemme, B.: Triangle-based obstacle-aware load balancing for massively multiplayer games. In NetGames (2011)
11. Dey, A.K., Understanding and using context. Personal and ubiquitous computing. 5(1):4-7 (2001)
12. Ducheneaut, N., Yee, N., Nickell, E., and Moore, R.J.: The life and death of online gaming communities: a look at guilds in world of warcraft. In SIGCHI, 839–848. ACM (2007)
13. Epure E., Hug C., Deneckère R., Brinkkemper S. What Shall I Do Next? Intention Mining for Flexible Process Enactment. 26th Int. Conf. CAiSE, Thessaloniki: Greece. Springer, 8484, pp.473-487, LNCS (2014)
14. Fayyad, U., Piatetsky-Shapiro, G., Smyth, P.: From data mining to knowledge discovery in databases. AI Magazine; 17(3): 37-54 (1996)
15. Gal, V., Le Prado, C. Natkin, S. and Vega, L.: Writing for video games. Proceedings Laval Virtual (IVRC) (2002)
16. Guo, Y, and Iosup, A.: The game trace archive. In Proceedings of the 11th Annual Workshop on Network and Systems Support for Games (NetGames '12). IEEE Press, Piscataway, NJ, USA (2012)

17. Harel, D., Segall, I., Kugler, H., and Setty,Y.: Crafting game-models using reactive system design. In Proceedings of the 2008 Conference on Future Play: Research, Play, Share, 121-128. ACM (2008)
18. Harel, D. and Marelly, H.: Come, let's play: Scenario-based programming using LSCs and the Play-Engine. Vol. 1. Springer Science & Business Media (2003)
19. Harel, D., Gery, E.: Executable object modeling with statecharts. in proceedings of the 18th International Conference on Software Engineering, 246-257, ICSE'96, IEEE Computer Society, Washington, DC, USA (1996)
20. Harel, D: Statecharts: A visual formalism for complex systems. Science of computer programming 8(3), 231-274 (1987)
21. Herbst, J.: A machine learning approach to workflow management. In Machine Learning: ECML 2000 (pp. 183-194). Springer Berlin Heidelberg (2000)
22. Hug, C., Deneckère, R., Salinesi, C.: Map-TBS: Map process enactment traces and analysis. Colette Rolland, Jaelson Castro, Oscar Pastor. International Conference on Research Challenges in Information Science (RCIS), Valencia, Spain. IEEE, 204-209 (2012)
23. IBM. An architectural blueprint for autonomic computing. Technical report, IBM (2003)
24. Jablonski, S. and Bussler, C.: Workow Management: Modeling Concepts, Architecture, and Implementation. International Thomson Computer Press (1996)
25. Jaffal, A., Le Grand, B, and Kirsch-Pinheiro, M.: Refinement Strategies for Correlating Context and User Behavior in Pervasive Information Systems. International Workshop on Big Data and Data Mining Challenges on IoT and Pervasive Systems, London (2015)
26. Jaffal, A., Kirsch-Pinheiro, M., Le Grand, B., Unified and Conceptual Context Analysis in Ubiquitous Environments. 8th International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, 48-55 (2014)
27. Khodabandelou, G. Mining Intentional Process Models, PhD Thesis, Université Paris 1 (2014)
28. Khodabandelou, G., Hug C., Salinesi C. A Novel Approach for Process Mining: Intentional Process Models Discovery. Eighth IEEE International Conference on Research Challenges in Information Science (RCIS), Marrakech: Morocco. pp. 1-12 (2014)
29. Kinicki, J., and Claypool, M.: Traffic analysis of avatars in Second Life. In NOSSDAV (2008)
30. Kourouthanassis, P.E., Giaglis, G.M., Vrehopoulos, A., Enhancing the user experience with pervasive information systems. International Journal of Information Management. 27: 319-335 (2008)
31. Kushnareva, E., Rychkova, I, and Le Grand, B.: Modeling Business Processes for Automated Crisis Management Support: Lessons Learned. n proceedings of: IEEE Ninth International Conference on Research Challenges in Information Science (RCIS'15), May 13-15 2015, Athens, Greece (to appear in 2015)
32. La Rosa, M., Dumas, M.,ter Hofstede, A.H.M., Mendling, J.: Configurable multi-perspective business process models. Preprint submitted to Information Systems Available at QUT Digital Repository: http://eprints.qut.edu.au/ (2009)
33. Ma, J., Zhang, W.X., Cai, S.: Variable threshold concept lattice and dependence space. Proceedings of the Third International Conference on Fuzzy Systems and Knowledge Discovery, 109-118 (2006)
34. Mostéfaoui, K., Pasquier-Rocha, J., Brézillon, P. : Context-aware computing: a guide for the pervasive computing community. Proceedings of the IEEE/ACS International Conference on Pervasive Services (IPCS'04), 39-48, IEEE Computer Society (2004)

35. Murata, T.: Petri nets: Properties, analysis and applications," in proceedings of the IEEE 77(4), 541-580 (1989)
36. Najar, S., Saidani, O., Kirsch-Pinheiro, M., Souveyet, C., Nurcan, S. Semantic representation of context models: a framework for analyzing and understanding. Proc. of the 1st Workshop on Context, information and ontologies, European Semantic Web Conference (2009)
37. Pensa, R.G., Leschi, C., Besson, J., Boulicaut, J.F.: Assessment of discretization techniques for relevant pattern discovery from gene expression data. 4th ACM SIGKDD Workshop on Data Mining in Bioinformatics, 24-30 (2004).
38. Plihon, V.: Un environnement pour l'ingénierie des méthodes. PhD thesis, Université Paris 1 Panthéon-Sorbonne (1996)
39. Plotkin, G.: A structural approach to operational semantics (1981)
40. Pohl, K., Weidenhaupt, K.: A contextual approach for process-integrated tools. In: Jazayeri, M., Schauer, H. (eds.) Software Engineering ESEC/FSE'97, LNCS, vol. 1301, 176-192. Springer Berlin Heidelberg (1997)
41. Priss U., Formal Concept Analysis in Information Science. In: Blaise, C. (ed.) Annual Review of Information Science and Technology, 40, 521-543 (2006)
42. Ralph P., Wand Y.: A Teleological Process Theory of Software Development, JAIS Theory Development Workshop, vol. 8, N°23 (2008)
43. Ramakrishnan, A., Preuveneers, D., Berbers Y., Enabling self-learning in dynamic and open IoT environments. 5th Int. Conference on Ambient Systems, Networks and Technologies (ANT-2014), Procedia Computer Science, Elsevier, 32: 207-214 (2014)
44. Rolland, C. : L'ingénierie des méthodes: une visite guidée. e-TI, 1 (2005)
45. Rolland, C., Prakash, N., and Benjamen, A.: A multi-model view of process modelling. Requirements Engineering, 4(4), 169-187(1999)
46. Rolland, C., Souveyet, C., Moreno, M.: An approach for de

ning ways-of-working. Information Systems 20(4), 337-359 (1995)
47. Rumbaugh, J., Jacobson, I., Booch, G.: Unifed Modeling Language Reference Manual, The (2Nd Edition). Pearson Higher Education (2004)
48. Russel, S. and Norvig, P.: Artificial Intelligence: A Modern Approach, 2nd ed. Prentice Hall (2003)
49. Rychkova, I., Le Grand, B., and Souveyet, C.: Towards executable specifications for Case Management Processes. Editors: Manfred Reichert, Roy Oberhauser, Gregor Grambow, Book Title: Advances in Ingelligent Process-Aware Information Systems (to appear in 2015)
50. Sawyer P., Mazo R., Diaz D., Salinesi C., and Hughes D.: Constraint Programming as a Means to Manage Configurations in Self-Adaptive Systems. Special Issue in IEEE Computer Journal "Dynamic Software Product Lines", ISSN 0018-9162, vol. 45, Number 10, 56-63 (2012)
51. Soffer, P., Yehezkel, T.: A state-based context-aware declarative process model. In: Enterprise, Business-Process and Information Systems Modeling, 148-162. Springer (2011)
52. Suznjevic, M., Stupar, I., and Matijasevic, M.: MMORPG player behavior model based on player action categories. In NetGames (2011)
53. van der Aalst, W.M.P, Weijters, A.J.M.M. and Maruster, L.: Workow Mining: Discovering Process Models from Event Logs. IEEE Trans. on Knowledge and Data Eng, 16(9), 1128-1142 (2004)
54. van Dongen, B. F., & van der Aalst, W. M. (2004). Multi-phase process mining: Building instance graphs. In Conceptual Modeling–ER 2004 (pp. 362-376). Springer Berlin Heidelberg.
55. Villalonga, C., M. Bauer, V. Huang, J. Bernat, et P. Barnaghi : Modeling of sensor data and context for the real world Internet. PerCom'10 Workshops Proceedings (2010)
56. Weijters, A.J.M.M. and van der Aalst, W.M.P.: Rediscovering Workflow Models from Event Based Data using Little Thumb. Integrated Computer-Aided Engineering, 10 (2), 151-162 (2003)
57. Wille, R., Formal Concept Analysis as Mathematical Theory of Concepts and Concept Hierarchies. In: B.Ganter et al., Formal Concept Analysis, 1-33. Springer-Verlag (2005)
58. Zaki, M. J. SPADE: An Efficient Algorithm for Mining Frequent Sequences, Machine Learning Journal, vol.42, pp.:31-60 (2001)