Available online at www.sciencedirect.com

ScienceDirect

Procedia Technology 11 (2013) 696 – 703

Procedia Technology

The 4th International Conference on Electrical Engineering and Informatics (ICEEI 2013)

Knowledge Representation and Inference Engine Model of SAPS Gaming Concept

Ririn Dwi Agustin*, Iping Supriana Suwardi, Ayu Purwarianti, Kridanto Surendro

STEI ITB , Ganeca 10 Bandung , West Java, Indonesia

Abstract

This paper propose a conceptual graph model to represent the concept of gaming, named SAPS (Status, Access, Power, Stuff) and propose a building block of the engine to run and control the game by rule of game design that be expressed in the SAPS. The advantage of this design is flexibility of the rule definition, supports many game mechanic and dynamic, as well as accommodates multirole playing and multiplayer games. The Excellence is achieved through representation of an access as actionID, Heuristic Value, Role name, Prerequisite, Impact. Prerequisite and impact are represented into a positive sentence of first order predicate calculus with the AND operator. The predicate in pre-requisite sentences express status and criteria. Predicate in impact have been identified 4 kinds, namely increment, decrement, add, and sets. Are those representative for update status, still need more research .Game mechanic was accommodated through status that be represented in (type of status; objects; attributes; criteria). Valid Type of status includes xpoint, redeemable Point, level/badge, progress, stuff, ownership, and time. Game Dynamic was accommodated through the discretion to perform user defined action. But the form of human computer interaction of such action has not been designed. Heuristic value on the model is filled by game designer and designed to support the delivery guidance to the player automatically, by suggesting optimal actions to achieve goals using Forward A* Search. Moreover three modules of Forward Intelligent Searching are designed, ie first to update the state of action based on current status. There are three kinds of state, namely the 'lock', 'open', 'closed'. The second is to update the status automatically based on the selected action to be executed. The third is execute automatic action that be triggered by status or by time. They all refer to the rule of the game are defined in the 'access'

Keywords: Gaming Concepts; SAPS, Knowledge Representation; Inference Engine; A* Algorithm

* Corresponding author.
E-mail address: ririn_dwia@unpas.ac.id

1. Introduction

SAPS (Status, Access, Power, Stuff) is a concept to make fun, engaging, and motivating game. SAPS is proposed by Gabe Zicherman[1], the guru of gamification. Status represents player position relative to the standard or to the other player. Status is used to recognize user effort or performance, so it must be designed carefully based on fun theory. Access is collection of rule that defines the rights of players to perform various activities and use a variety of services in the game. Powers granted to players who have a certain status in order to facilitate the other players are involved in the same game. This is used for gain virality impact of gaming. Stuff is something or free facilities granted to the player as a reward for achievement of their performance.

MDA (Mechanics, Dyanmics, Aesthetic) is a framework to develop and research in game. This framework is hoped to bridge between game design process, game development, game criticism, and technical game research [2]. According MDA, game can be view as three, those are RULE-SYSTEM-FUN and then be transformed to MECHANICS, DYNAMIC, AESTHETIC. Mechanics aspect can be understood as tools for playing, dynamic is define about how interaction between player and tools. Both of them is used to create aesthetic or FUN of game.

SAPS can combine in MDA Framework. SAPS give simple and powerful concept to handle and manage relation between game mechanic and dynamic. Game mechanic is more closed to status and game dynamic is closed with access. Pragmatically acces declare about relation between action and status. Kind of action defined by varian of game dynamic.

Based on experience in previous paper about “Using Gamification in Design of Application S/W for Final Project Course Management “[3], SAPS can be represented as a graph (see figure 1). Node in the graph represents action or status. Edge represents relation with them. With node that represents action, “In Edge” is meaning pre-requisite while “out edge” is meaning impact.

For proceed to the construction phase of our gamification that being studied, it would require the design of data structures and programs that are flexible and robust. Through this paper, we presented the results of the SAPS construction design combined with dynamic games and game mechanic of the MDA framework. The design was named model of conceptual graph to represent the concept of gaming called SAPS (Status, Access, Power, Stuff) and a building block of the engine to run and control the game by rule of game design expressed in the SAPS.

Exposure the idea begins with the conceptual graph models, the details of each element of the data structure model, followed by describe of the proposed architecture engines and engine operation pseudo code algorithms, and concludes with examples of the representation with a case study on the gamification of Final Project Course Management at Informatic Engineering Pasundan University.

2. Graph conceptual model of SAPS

For Fig. 1, can be understood that student only can take “learn” or “registration” for the first time, because only those status did not have “in arrow”. “Learn” is action to get “knowledge level” higher. Specific knowledge level was needed for “manage artifact” action. The Goal of this game is getting the highest status or badge that can be reached through core activity. Player has to have some requirement and adjusts with specifics schedule to take the activity. There are two kinds of activity, the first is real bussines process activity and second is gaming activity. (1) Learn, (2) Work, (3) Buy Golden Ticket (4) Give The Golden Ticket to friend (5) Take Red Carpet Service (6) Give/unlock badge are gaming activity. The rests are activity from organization business process. Unlock badge is automated action by system.

Graph conceptual model was designed based on SAPS concept and aspect of real world that be made gamification. Nodes on the conceptual graph models (see fig 2) are divided into two types, namely status and action. Status is symbolized by circles and action is symbolized by square. In terms SAPS, action does not appear explicitly, however actually action is inside A (access).
Action classified into 3 types, namely

- provided for be chosen by the player and only have an impact on himself,
- provided for be chosen by the player and the impact on other players (POWER)
- executed by the system automatically based on the specific conditions of status, stuff or based on time. Action

like this is usually used to give a reward or a punishment to the player.

Node status can be divided into three kinds, namely status, stuff, and time.
- Status to accommodate a variety of game mechanic used in gaming, but still bounded domain of type set and integer.
- Stuff can be treated the same as the status, but it is intentionally separate node because it is one identifier SAPS. Besides, it is also recommended to use stuff like this for the sake of managing POWER.
- Time is a status that has special characteristics and level of interest. Player is not allowed to perform actions that have an impact on the 'time'. Incremental fixed in time by the system is managed by a separate game. Game designers can define a game mechanic that is a derivative of the time, such as the schedule, appointments or countdown.

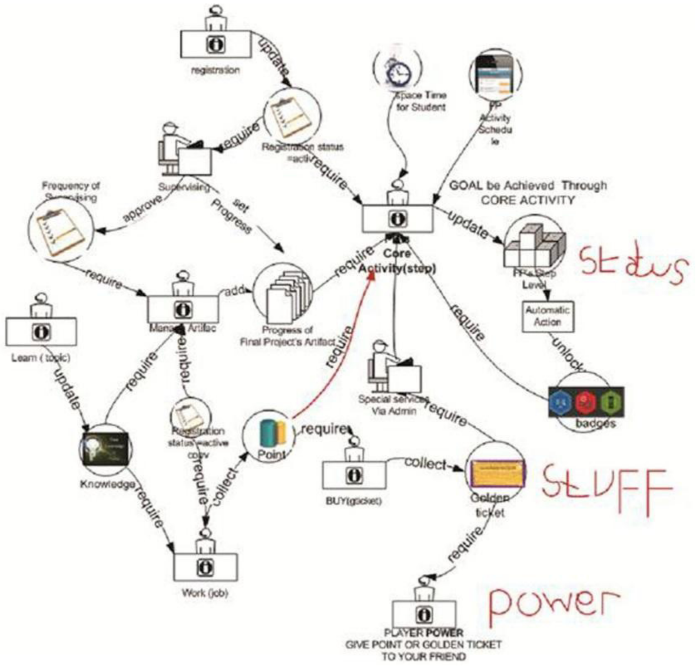

Fig. 1. Game Concept Design of Gamification in Final Project Course Management at Informatics Engineering Pasundan University Improved from [3]

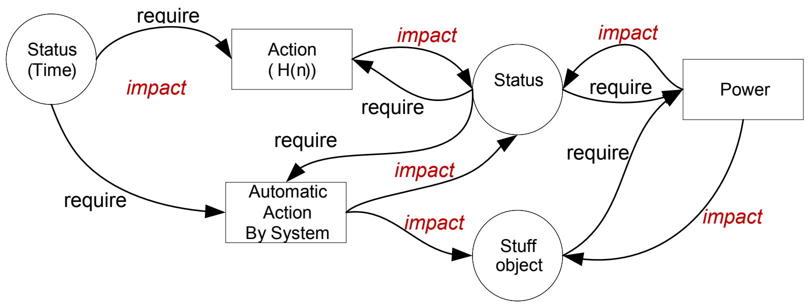

Fig. 2. Graph Conceptual Model of SAPS

```
Action (name,typeofaction,heuristicValue,object(O-A-V))
Accesed by
   Type Of role
Require
   Listof (TypeOfStatus,name,nameOfAtribut, criteria)),
Impact
 Listof(Decrement(TypeOfStatus,name,nameOfAtribut,constant)),
 Listof(Increment(TypeOfStatus,name,nameOfAtribut,constant),
 Listof (Set (TypeOfStatus,name,nameOfAtribut,  constant)),
 Listof (Add (TypeOfStatus, name,nameOfAtribut, constant)).

TypeofAction = request|automaticByStatus|automaticByTime
TypeOfStatus= Xpoint|RdPoint|Progress|Level|Stuff|Time
```

Fig. 3. Data Structure of SAPS's Graph Conceptual Model

The advantage of this design is flexibility of the rule definition, supports many game mechanic and dynamic, as well as accommodates multirole playing and multiplayer games. The Excellence is achieved through representation of an access as actionID, Heuristic Value, Role name, Prerequisite, Impact.

Prerequisite and impact are represented into a positive sentence of first order predicate calculus with the AND operator. The predicate in pre-requisite sentences express status and criteria. Predicate in impact permitted only 3 kinds, namely increment, decrement, and sets. Those are representative for update status.

Game mechanic accommodated through common status that be represented in (type of status; name of objects; attributes; criteria). Valid Type of status includes xpoint, RdeemablePoint, level, Progress, Stuff, ownership and Time. These are brief of the variant of common status.

- Xpoint is status that represents experience of player. This status always increase cannot decrease but can expire. Action with greater Xpoint needed mean more important than the others.
- RdPoint is stand for Redeemable point. This point can be barter with special object via virtual economy in game. Rpoint can be get by player through the action taken. More valuable action more Rpoint that be given.
- Level is indicating milestone progress or achievement of the player in game over time. It can be used as marker and mean that another aspect in game different than another level. Example, player gets more Rdeemable point by same action if they are at different level. Level can express by badge for reputation.
- Progress is indicating detail increase of player performance; indicate by number 0 … 100%.
- Collection/ownership is represent object that be collected by player
- Stuff is something or free facilities granted to the player as a reward for achievement of their performance. Stuff can be treated as common status.
- Time is important aspect in game. In this paper basic time is managed only by fixed decrement or increment. The basic time component is used for another game mechanic, like appointment , countdown, reward schedule

Relationship between detail of action and variant of common status be described at Fig. 4.

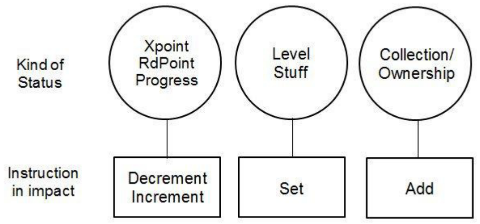

Fig. 4. Relationship Kind of status v.s instruction of Impact

Game Dynamic accommodated through the discretion to perform user defined action. But the model of human computer interaction of such action has not been defined in this paper.

Heuristic value on the model is filled by game designer and designed to support the delivery guidance to the player automatically, by suggesting optimal actions to achieve goals. Although for the sake of providing a "FUN GAME", the player only provided advice one step ahead. Searching principle used is Forward A * Search with the proposed

H1 (n) = Number of minimal action is still required to reach the goal or

H2 (n) = estimated total Redeemable Point and Point Experience needed to achieve goals.

While G (n) = total issued Redeemable Point - Total Redeemable Point earned.

Moreover three modules of Intelligent Searching are required, ie first to update the state of action based on current status. There are three kinds of state, namely the 'lock', 'open', 'closed'. And the second is to update the status automatically based on the selected action to be executed. Both refer to the rule of the game are defined in the 'acces'

3. Architecture of inference engine

Knowledge base system or reasoning system is built from two main components. Knowledge base is component that domain specific content and inference engine is component that domain independent algorithm. First order predicate calculus is more universal knowledge representation than the others and have been proven can be implemented well. The idea behind this architecture (see Fig. 5.) is that concept.

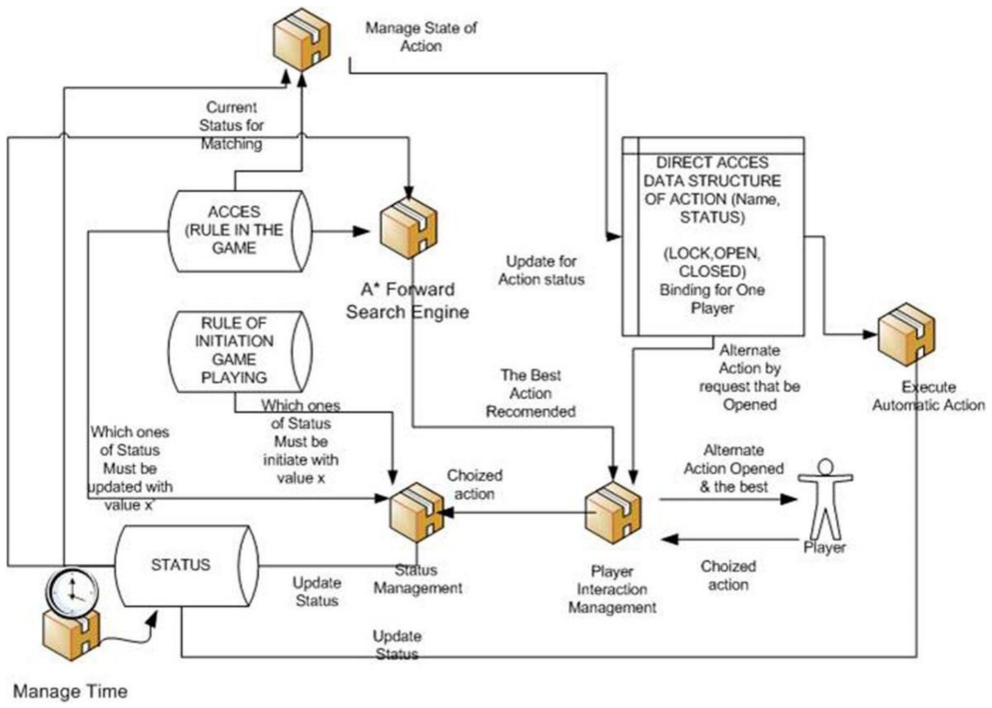
Manage State of Action
Current Status for Matching
ACCES (RULE IN THE GAME)
A* Forward Search Engine
RULE OF INITIATION GAME PLAYING
DIRECT ACCES DATA STRUCTURE OF ACTION (Name, STATUS)
(LOCK,OPEN, CLOSED) Binding for One Player
Update for Action status
Alternate Action by request that be Opened
Execute Automatic Action
The Best Action Recomended
Which ones of Status Must be initiate with value x
Which ones of Status Must be updated with value x
Choized action
Alternate Action Opened & the best
Player
Choized action
STATUS
Update Status
Status Management
Player Interaction Management
Update Status
Manage Time

Fig. 5. SAPS Inference Engine Architecture

Pseudocode of SAPS Inference Engine

```
{Player Start}
        Initiation Status of Player
        Initiation Action State in Array Of Action
        Initiation Time

Loop Until Stop
        Repeat
                ActionState-Management(ActionState,current_ status,Acces)
                Execute automatic Action
        Until (no actionState change because automatic action)
        Best_action ← A* Search(current_ status, Acces)
        If player make request Then Give_advis(Best_action)
        Get_player_action(choized_action)
        Status_Management(choized_action,Access)
        Update(time)
```

4. Example of representation

Graph in Fig. 4, is detail SAPS of core activity of gamification in final project course management. It is derived from graph at fig 1, doing to show the goal stated and the new problem. The goal state is “completed” and then gets “the blue badge”. This goal is needed to calculate H(n). The new problem that be found is OR relationship between “in edge” to action.

These are different case of action to be be exemplified :(1) normal action for player = student, (2)Normal action for player = supervisor , (3)Normal action for player = manager , (4)Action with OR of “in Edge”, (5)Automatic action, (6)Action with OR of impact

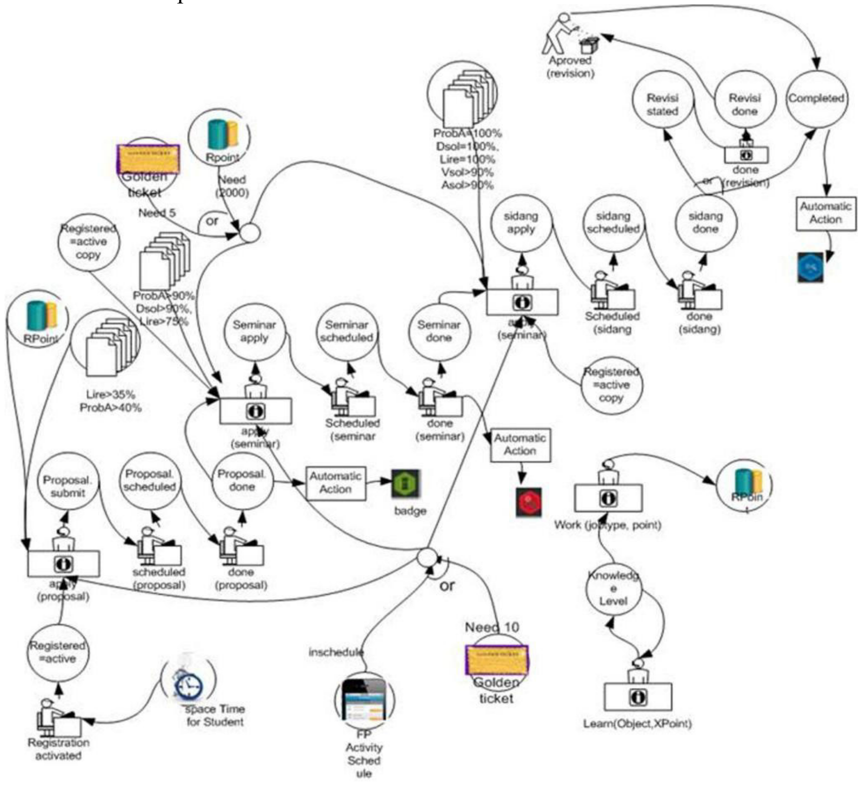

Fig. 6. Detail of Core Activity Final Project Business Process (derived from Fig 1)

1. Normal action for player =student

```
Action (apply,request,11,object(proposal,studentID,X))
Accesed by
       student
Require
       ((Level, registration, , “=’active’”),( RPoint, work ,saldo , “>2000”),
       ( Time, schedule,proposal,”in”),( progress,artifak,lire,”>=35%”),/*literature review*/
       ( progress,artifak,proba,”>=40%”)) /* problem analysis*/
Impact
        ((Decrement (Rpoints,Work,saldo,2000)),(Set (level,proposal,,”submit”))).
```

2. Normal Action with player = supervisor

```
Action (aproved,request,1, object(revision, studentID,X))
Accesed by
       supervisor
Require
       ( Level,revisi,,”=’done’”)
Impact
       ((Set (level,revisi,,”completed”))
```

3. Normal Action with player = manager

```
Action (done,request,5, object(seminar, studentID,X))
Accesed by
       manager
Require
       ((Level, seminar, , “=’scheduled’”))
Impact
       (Set (level,seminar,,”done”))
```

4. Action with OR at “in edge”

```
   a. Action (apply,request,8,object(seminar,studentID,X))
      Accesed by
             student
      Require
      ((Level, registration, , “=’active’”),( RPoint, work ,saldo , “>2000”),
      ( Time, schedule,proposal,”in”),( progress,artifak,lire,”>=75%”),
      ( progress,artifak,proba,”>=90%”, ( progress,artifak,Dsol,”>=90%”))
      Impact
       (Decrement (Rpoints,Work,saldo,2000)),(Set (level,seminar,,”submit”))
   b. Action (apply,request,8,object(seminar,studentID,X))
      Accesed by
             student
      Require
      ((Level, registration, , “=’active’”),( Stuff, goldenticket , , “>=5”),
      ( Time, schedule,proposal,”in”),( progress,artifak,lire,”>=75%”),
      ( progress,artifak,proba,”>=90%”, ( progress,artifak,Dsol,”>=90%”))
      Impact
       (Decrement(Stuff,goldenticket,,5)),(Set (level,seminar,,”submit”))
   c. Action (apply,request,8,object(seminar,studentID,X))
      Accesed by
             student
      Require
      ((Level, registration, , “=’active’”),( RPoint, work ,saldo , “>2000”),
      ( progress,artifak,lire,”>=75%”),(stuff,goldenticket,”>=10”)
      ( progress,artifak,proba,”>=90%”, ( progress,artifak,Dsol,”>=90%”))
      Impact
       (Decrement (Rpoints,Work,saldo,2000),(stuff,goldenticket,,10),
       (Set (level,seminar,,”submit”))
   d. Action (apply,request,8,object(seminar,studentID,X))
      Accesed by
             student
      Require
      ((Level, registration, , “=’active’”),( Stuff, goldenticket , , “>=15”),
      ( progress,artifak,lire,”>=75%”),
      ( progress,artifak,proba,”>=90%”, ( progress,artifak,Dsol,”>=90%”))
      Impact
       (Decrement(Stuff,goldenticket,,15)),
       (Set (level,seminar,,”submit”))
```

5. Automatic Action

```
Action (givebadge,automatic,5, object(green, studentID,X))
Accesed by
        "system"
Require
   ((Level, proposal, , "='done'"))
Impact
   (Set(level,badge,,"greed")),
   (modify (Action (givebadge,automatic,5, object(green, studentID,X),"closed").
```

6. Action with Or at it Impact
   Data structure in this model have not yet supported this condition. This problem can be solved with make two actions with different prerequisite.

5. Conclusion

Flexibility of this model has been described at example of representation and relationship between game mechanic and instruction at impact section in the representation of graph conceptual model. If game developer need special impact related with new game mechanic then the instruction can impact be added in simple way. Robustness can be seen at using the proven concept those are graph conceptual and horn clauses of first order predicate calculus. Computation theory has had many proven algorithm to solve anything problem over the representation.

Nevertheless further research is still needed to test the coding level and appropriateness of this design on another gaming case.

References

[1] Zicherman, Gabe & Cunningham, Christopher. Gamification by Design: Implementing Game Mechanics in Web and Mobile Apps. Canada: O'reilly;2011.
[2] Hunicke,Robin, LeBlanc Marc, Zubeck Robert. MDA: A Formal Approach to Game Design and Game Research, CiteSeerX: 10.1.1.79.4561; 2004.
[3] Agustin, Ririn Dwi ,Suwardi, Iping Supriana, Purwarianti, Ayu, Kridanto, Surendro. Using Gamification in Desain of S/W Aplication for Final Project Course Management Case Study at Informatics Engineering Pasundan University , Proceeding of ICIBA; 2013.