Autonomous Systems Project - Notes

Roberto Casadei
roberto.casadei12@studio.unibo.it

December 16, 2014

Contents

Chapter 1

Subject

1.1 Two Rooms and a Boom

1.1.1 WHAT: The Game

The short story.

***Two Rooms and a Boom* is a competitive, team-based, social deduction party game** set in two rooms where hostages are exchanged at each round. The (initially mutually extraneous) players in the rooms interact to ultimately allow the Bomber to assassinate the President or the latter to remain in life.

The long story.

Two Rooms and a Boom is a competitive, social deduction party game. There are two teams: the Red Team and the Blue Team. Players are secretely given a card that certify their team and role: this is the sole information a player knows at the beginning of the game[1]. There are two special roles: the *President* (in the Blue Team), and the *Bomber* (in the Red Team). Players are equally distributed between two rooms (i.e., separate playing areas). Then, the game consists of many timed rounds. In each round, players interact with each other by exchanging information. At the end of each round, some players (i.e., the hostages) will be swapped into opposing rooms, as decided by the room's leader that had been elected for that round. If the Red Team's Bomber is in the same room as the President at the end of the game, then the Red Team wins; otherwise the Blue Team wins. Lying encouraged.

[1] In addition to the rules of the game, of course.

1.1.2 Why is this game funny?

***Two Rooms and a Boom* entertains its players by making them interact in a *socially dynamic context* where *competition*, *cooperation*, and *unpredictability* play a key role**. People stay puzzled: "is he a team-mate or an enemy?", "what's her strategy?", "what's the deep reason for his actions?", "what's happening in the other room?", "what happens if I do this?"

Interaction has *no rules*; the only regulations that apply are those of your national legal system: players can lie, break agreements, remain silent, mix truth and falsehood, steal one's card, and so on.

Moreover, *the fear of being discovered, the cleverness of the competition, the unexpected and the hazard stimulate emotions such as excitement, fear, pleasure and reward.*

Other characteristics that might contribute to the comical dimension follows:

- The game is *short* and doesn't require complex setups
- Each execution is *potentially different* from the others
- The game is also very *extensible/customizable* to support a variety of new situations or to adapt it to generate more enjoyable scenarios

1.1.3 Why is this game relevant within the Autonomous Systems course?

Basically, as many other games, ***Two Rooms and a Boom* builds on the *autonomy* of its actors (players) to be interesting and enjoyable**. In fact, a good player should be capable of *self-government* and self-protection against the *influence* of other players (which might try to manipulate it). Note that one player's autonomy does not just need to be preserved against opponents, but also against team-mates which might suggest uneffective strategies or communicate biased information.

Good players should be *intelligent* and, in particular, capable of complex *epistemic reasoning*, based on all the knowledge they hold about the game. Note that the game would be quite boring if a player made deterministic decisions exclusively based on certain knowledge. So, the game is interesting because the players can take *unpredictable* decisions and because they can elaborate effective strategies based on *probabilistic, multi-level reasoning.*

Also, it's important for a player to be able to *predict* (with a certain *confidence*) the behavior of the other players. In doing so, one could reason by asking himself: "what are their *intentions*? what do they *believe* about me?" (cf. intentional stance).

Another reason for which players should have strong *executive* autonomy[2] and intelligence lies in the fact that each player is *responsible* in respect to all the team-mates. That is, any "stupid" action might offend the entire team.

The competition is ruled so as to foster two *conflicting (sub-)goals*. As information potentially represent a competitive advantage, on one side, the less information is given to opponents, the lower will be their ability to take profitable decisions. On the other side, the more information is spread across the team, the better chance to perform well.

When a player discovers one or more of its team-mates, things get interesting as they can exchange information and *cooperatively* build strategies. In particular, a *team-level strategy* could be defined (e.g., "if you'll find yourself in this situation, do X and Y, and I'll do W and Z").

Imagination is also important to *predict* future situations (e.g., "what is happening in the other room?" or "what will happen in the next round?"). Humans in the real game may also have *intuitions* ("*sixth sense*") or capture *unexplicit* signs (e.g., based on personal acquaintance with the other players).

In a nutshell, **this game exhibits a complexity and characteristics that make it amenable for analysis/modelling/engineering within the conceptual framework developed in the context of the Autonomous Systems course**.

[2] Note that nothing prevents the players to exhibit *motivational* autonomy as commonly the actual (meta-)goal – in most of the games – is to simply enjoy.

1.2 Analysis

1.2.1 Elements

- 2 Teams: blues and reds
- N Players per team
- 2 Rooms
- Team roles
  - President (only 1 in blue team)
  - Bomber (only 1 in red team)
  - Normal player (N-1 per team)
- Roles within a room
  - Normal actor (N-1 per room)
  - Room leader (1 per room)
- Actions (see Section 1.2.2)
- Information units
  - Team/role of a player
  - Strategy (i.e., mapping from contexts to actions?) of a player
  - Statements by other players
  - Observed actions in the past
  - ...Opinions? Confidence about one's future actions? Confidence about future states-of-affairs?...
- R rounds
- Phases
  1. Setup: assignment of a pair (team, role) for each member; equally-sized random distribution of players into the two rooms
  2. Round
     2.1 Room leader selection
     2.2 Interaction
     2.3 Exchange of hostages (as decided by the room leaders)
  3. End of game or back to Phase 2 if $num\_rounds < R$
- Rules
  - Room leader selection by voting (if parity, a leader is randomly chosen)
  - The players cannot interact during leader selection
  - Reds win if the Bomber is in the same room as the President; otherwise, blues win.
  - A round terminates once a *round termination condition* takes place (e.g., time-based or per num-of-interactions)
  - (No rules in interaction)

1.2.2 Actions

An *action* is any interaction, performed by an a player, which potentially has an effect on the world, that is, on the other players or on the game.

There are two kinds of actions:

1. Communication actions
2. Game-related actions

While communication actions can be performed in any game phase, other actions are specific to a given phase. So, doing actions at the wrong time will simply make them fail.

Phase or context-specific actions include:

- Room leader selection
  - Vote
- Interaction
  - Co-reveal
  - Colour-reveal
- Hostages exchange
  - (Only for room leaders) Selection of hostages

Communication actions

The game is very free for what concerns the interactions that can take place.

However, typical communications include:

- General
  - Ask X for some information
  - Tell some information to X
- Interaction
  - Ask X to co-/colour-reveal
  - Accept/reject to co-/colour-reveal

1.2.3 Reasoning

Typically, complex practical and epistemic reasoning is carried out by players.

The reasoning process has the following characteristics:

- It may have "genetically" specific features – i.e., each player may reason in a different manner or have different *attitudes*
- It is probabilistic/stochastic

The following are examples of reasoning.

Epistemic reasoning

Given	Inferred knowledge
(i) The number of players (ii) The distribution of roles in a team (iii) The roles of a subset of players	The probability of player P to have role R
(i) Player X asks me to co-reveal (ii) I don't know his/her role	`X = president/bomber` is quite probable.
(i) A team-mate tells me some information	That information is true.
...	...

Practical reasoning

Given	Inferred action
(i) Me and my team-mates have decided that X has to be voted	Vote X
(i) I am the room leader and a red player (ii) It is the last round (iii) The bomber is in this room (iv) The president is in the other room	Send the bomber to the other room based on the probability of the president to be sent here
...	...

1.3 Technical requirements

The analysis pointed out many features of the game.

When the game is implemented as a software system, it should satisfy the following technical requirements:

- **Ontological reasoning**: so that agents can infer implicit knowledge from explicitly represented facts
- **Probabilistic reasoning**: so that agents can deal with uncertainty

In addition to such aspects, we should be able to express the following features:

- **Attitudes** for agents
  - E.g., the *"character"* (inclination) of a player may be considered as associated with a particular *"approach"* to the game (which ultimately defines a coherent position wrt strategies). However, players may occasionally take *unexpected* courses of action.
- **Cooperative strategies**
- **Stochastic behavior**
- (**Falsehood** and **trust**?)

1.4 Feasibility analysis

A question that must be addressed is: **can we build, with *reasonable* effort, a computer program that is able to play this game *effectively*?**

Of course, the answer depends on the meaning of "reasonable" and "effectively".

At a first sight, *Two Rooms and a Boom* is a game that just *seems* unplayable by computers[3].

Human players try to do their best to conceal information, misguide supposed enemies, perturb other players, deduce facts from any manifestation or sign (be it verbal *or not*), or elaborate strategies that could not be predicted by other players. Very complex reasoning takes place, clearly multi-level (it's common to reason about the way other players reason) and based on a lot of context information.

In general, multiple levels of feasibility can be considered:

1. *Theoretical feasibility of reasoning*: relating to decidability issues in ontological/probabilistic reasoning
2. *Practical feasibility of reasoning*: relating to complexity and time constraints
3. *"Turing-test" feasibility*: can we build agents that are able to play this game just like humans do?
4. *Project feasibility*: can we build a system of acceptable quality within reasonable time/effort constraints?

The "plain vanilla" game seems to be too unconstrained (free of rules). As a consequence, the game designers have the burden to codify anything the players can talk about.

So, we should think about the possibility of shaping/restricting the game so as to find a *trade-off* between complexity and playability.

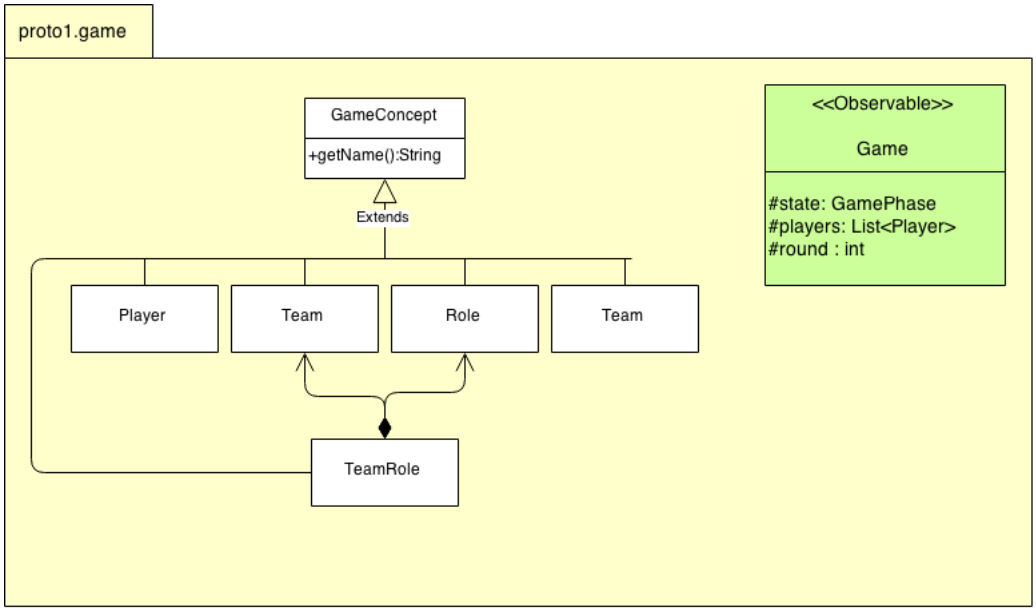
[3] "You insist that there is something that a machine can't do. If you will tell me precisely what it is that a machine cannot do, then I can always make a machine which will do just that." (Von Neumann)

1.5 Prototyping

1.5.1 Game modelling

The object-oriented paradigm effectively supports the modelling of static game elements and game-related information.

The following UML diagram concisely shows these entities by a *structural* point of view.



1.5.2 Mapping from game to concepts

Players – First of all, they are **autonomous** entities. Secondly, they play the game, they do something, they *act*. Thus, players can be naturally mapped to **agents**. They are autonomous and, as a consequence, their *moving force* is internal; i.e., they are **proactive**.

```
/* Initial goals */

!start.

/* Plans */

+!start <- wanna_play.
```

Actions – The actions are *situated* in the sense of Suchman [1]. In other words, the *context* in which they are carried out is prominent. A player could (as it is autonomous, "free") advance a vote for its candidate for room leadership during the phase of hostages swapping, but it would not have any effect as the action would be performed at the wrong time.

```
    @Override
    public boolean executeAction(String agName, Structure action) {
        boolean result = false;

        List<Term> terms = action.getTerms();
        String functor = action.getFunctor();
        Player p = game.getPlayerFromName(agName);

        if(functor.equals(Actions.WANNA_PLAY) && game.IsAt(GamePhase.Init) ){
            result = game.AddPlayer(agName);
        } else if(functor.equals(Actions.VOTE_LEADER) && game.IsAt(GamePhase.LeaderSelection)){
            String candidateLeader = ((Atom)terms.get(0)).toString();
            result = game.Vote(agName, candidateLeader);
        } else if(functor.equals(Actions.SELECT_HOSTAGE) && game.IsAt(GamePhase.HostageSelection)){
            String hostage = ((Atom)terms.get(0)).toString();
```

```
            result = game.SelectHostage(p, hostage);
        }

        /* ... */
    }
```

Two rooms – The players are **situated** in a room, and can be **moved** from a room to another. The rooms constitutes the game **environment** and effectively define a notion of **locality**, thus ensuring that only players of the same room can interact.

Game, game rules, game phases – The game phases set the **context** for the behavior of the agents.

```
+phase(leader_selection) <-
    !decide_who_to_vote(Who);
    vote(Who);
    .

+phase(interaction) <-
    /* ... */
    .

+phase(hostages_exchange) : .my_name(Me) & room_leader(Me) <-
    !choose_hostage(H);
    select_hostage(H);
    .
```

The players **coordinate** and act on the basis of the game rules. The environment, by representing the game in itself, can support the coordination of players in a **stigmergic** fashion.

Each agent keeps a **representation** of the state of the game and its *history* as well.

Hostages swapping – The fact that players can only be moved (and cannot freely move) should not be interpreted as a lack of autonomy ; instead, it should be thought as a *deliberate* reduction of self-determination resulting from the will of playing the game according to its rules. Then, player agents can be aware of that (e.g., by explicitly having the goal of playing the game) or not (e.g., by having the goal of following the rules of the game structurally embedded).

***Two Rooms and a Boom* as a *social* game** – The players in the game can be collectively mapped to the notion of **society**. As in human societies, the agents have different **roles** within a given social context (here, the game). As in human societies, different players may have conflicting **goals**. As in human societies, players with the same roles could **collaborate** to reach goals that could be hard to be achieved with isolated effort.

1.5.3 Implementation in Jason

Jason is a Java-based, multi-agent system development platform.

It has been chosen as the platform for the development of the prototype of the game because it provides many first-class abstractions that directly map to the elements of our conceptual game model.

The system

– The system (environment and agents) is declaratively described as follows:

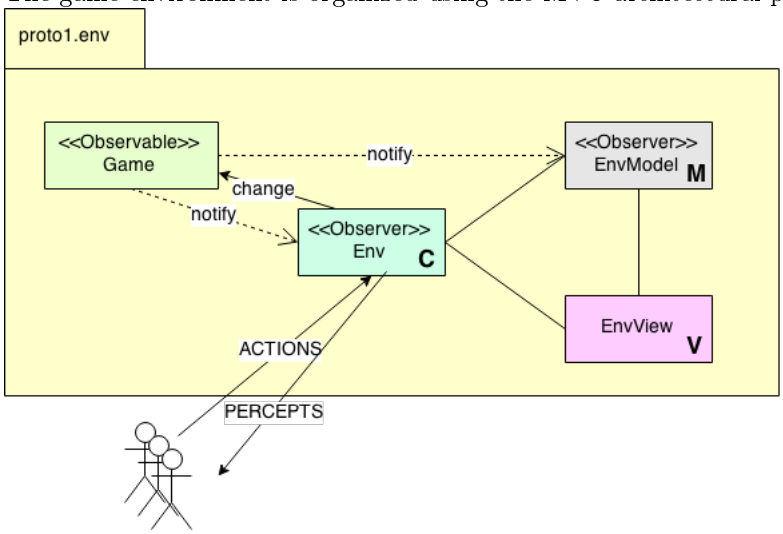
```
MAS prototype_TwoRoomsAndABoom {

    infrastructure: Centralised

    environment: proto1.env.Env(6, gui)

    agents:
        roby    proto1_dumb_player;
        marco   proto1_player;
        ste     proto1_player;
        fede    proto1_player;
        fanny   proto1_player;
```

```
13          vale    proto1_player;
14
15      aslSourcePath:
16          "src/asl";
17  }
```

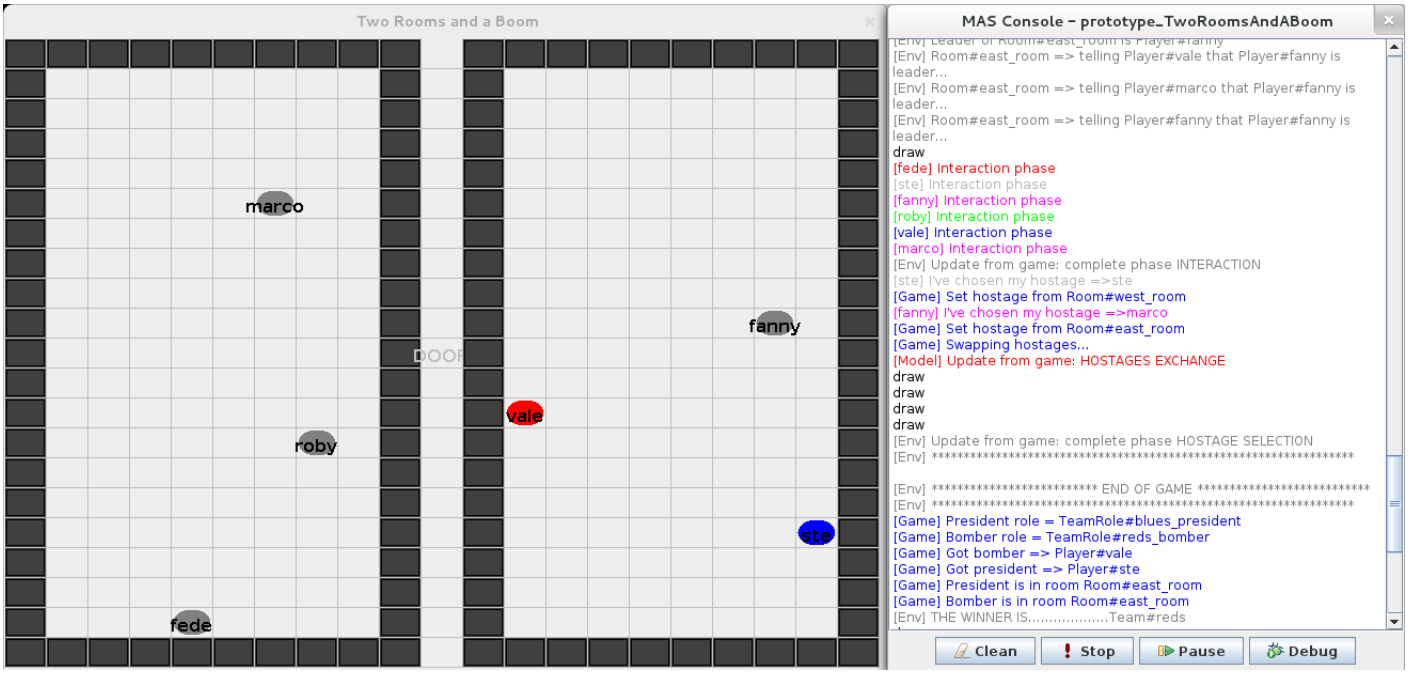
Architecture: components, organization, responsibilities

–

The game environment is organized using the MVC architectural pattern.



1.5.4 Screenshot



Bibliography

[1] L. A. Suchman, *Plans and Situated Actions: The Problem of Human-machine Communication*. New York, NY, USA: Cambridge University Press, 1987. 1.5.2