PLATO: A Coordination Framework for Designers of Multi-Player Real-Time Digital Games

Ariyan Zohoorian, Kevin G. Stanley, Carl Gutwin, Amin Tavassolian
Department of Computer Science
University of Saskatchewan
Saskatoon, Canada
firstname.lastname@usask.ca

ABSTRACT
Player coordination is a key element in many multi-player real-time digital games, and control over the design of these coordination requirements is an important part of developing successful games. However, it is currently difficult to describe or analyze coordination requirements in game situations, because current frameworks and theories do not mesh with the realities of video game design. We developed a new framework (called PLATO) that can help game designers understand and manipulate coordination episodes. Our framework deals with five atomic aspects of coordinated activity: Players, Locations, Actions, Time, and Objects. PLATO provides a vocabulary, methodology and diagram notation for describing and analyzing coordination. We demonstrate the framework’s utility by describing coordination situations from existing games, showing how PLATO can be used to understand and redesign coordination requirements.

Categories and Subject Descriptors
H.5.3 [Information Interfaces and Presentation]: Group and Organization Interfaces – Computer-supported cooperative work; K.8 [Personal Computing]: Games.

General Terms
Design, Human Factors, Theory

Keywords
Group coordination, real-time multi-player games.

1. INTRODUCTION
Cooperative real-time multi-player modes are now common in many digital-game genres, including first-person shooters (e.g., Call of Duty), platformers (e.g., LittleBigPlanet), puzzle games (e.g., Portal 2), and sports games (e.g., FIFA Soccer). In these games, players must coordinate their actions to achieve goals and objectives. In a shooter, for example, one player might have to throw a grenade into a room at the exact moment that another player kicks open the door, while a third player draws enemy fire. Similarly, players in Portal 2 co-op must carefully synchronize the placement of portals and objects to escape from a maniacal AI opponent. Coordination is an important part of the design of these games, for several reasons: different types and degrees of coordination can make the game easier or more difficult; coordination is an important part of ‘playing like a team’, a skill that can provide better players an advantage; and coordination is a shared activity that can add to the sociability of the game.

Being able to exercise control over the coordination elements that make up a multi-player game is an important part of game design, but it is currently difficult to describe, analyze, or evaluate coordination scenarios in games. Current frameworks and theories of coordination (e.g., Malone and Crowston [13] or Eccles [4]) are either not well matched to the resources and actions in video games, or are designed for slower collaborative processes rather than real-time activities. Similarly, current tools for diagramming coordination (e.g., Gantt or PERT charts) do not adequately specify the elements present in game situations. As a result, coordination design becomes an ad-hoc process, and designers may lose track of the details and complexities within their game’s coordination scenarios. Although playtesting can eventually uncover most problems in the coordination design, it would be useful to have a way to discuss and analyze coordination at earlier stages in the evolution of a multi-player game, as well as a common vocabulary and notation for analyzing coordination errors and opportunities discovered during playtesting.

To address these limitations and provide improved understanding of, and control over, coordination design in video games, we developed the PLATO framework, named for the five game elements that it encompasses: Players, Locations, Actions, Time, and Objects. PLATO provides two main constructs to designers: a set of core concepts and a vocabulary for discussing, describing, and analyzing coordination scenarios; and a diagram notation for visualizing coordinated activities. The understanding provided by the PLATO framework allows designers to better control the coordination requirements in their games, and adjust the type and difficulty of shared activities.

In this paper we introduce the PLATO framework, and show how our ideas can be used by designers. We demonstrate the expressive and descriptive power of the framework through a series of worked examples and a case study of coordination scenarios in Portal 2’s co-op mode. We make three main contributions: we extend ideas introduced by earlier theories of coordination to the domain of real-time multi-player games; we provide a new set of concepts that can be used to describe and characterize multi-player interactions; and we provide tools that can help designers produce better multi-player games through better control over coordination – tools that can work even at early stages of design.

2. RELATED LITERATURE
Coordination – the management of dependencies in the actions of two or more people – is a part of most shared activities, and
Researchers in several fields have studied group coordination (e.g., [1,3,7]). Researchers have looked at coordination in contexts as diverse as ship navigation [1], surgery [20,23], command and control [11], and shared-workspace groupware [8]. These studies show that group work has numerous episodes of coordination both at large time scales where planning and division of labour are primary activities (e.g., industrial or manufacturing processes [21]), and at the scale of individual actions such as passing a tool from one person to another, or timing an interruption to coincide with a break in another person’s activity (e.g., [8,10]).

One main theoretical approach to coordination is that of Malone and Crowston [13], who decompose collaborative activity into four constituent elements: Goals, Activities, Actors and Interdependencies. Interdependency among the activities creates requirements for coordination, and this theory identifies three kinds of interdependency: prerequisites, shared resources, and simultaneity. This framework was originally focused on large-scale work practices such as manufacturing, but has since been applied in other contexts as well.

A second view involves the idea of articulation work, which focuses on how coordination of complex activities occurs: “how cooperating actors devise and use coordinative constructs such as coordinative protocols and workflows and how such constructs are supported by artifacts” ([21] p.2). As discussed by Schmidt and Simone [21], articulation in larger-scale cooperation is often mediated by artifacts used at coordination junctures.

A third area of research considers awareness support as a necessary tool for enabling coordination, particularly in distributed groupware [1]. Researchers have looked at awareness requirements at several time scales, including traces of activity over long time periods [9] and real-time awareness of immediate actions [27]. Despite this work on real-time awareness, there is little work in on coordination requirements when teamwork happens quickly and when there is insufficient time to coordinate verbally. One exception is Hutchins’ study of ship-navigation teams [12]: in this context, time pressure (and severe consequences) forces the activity into a regimented structure.

Several researchers have also looked at tracing and characterizing interaction in collaborative learning systems. Prior work has visualized and described action logs using diagrams [e.g., 26,18] or regular expressions [18]. Researchers have investigated different visualization methods such as dependency graphs [22] to study interactional construction of meaning between two collaborators.

Researchers in kinesiology have more frequently examined the area of real-time coordination in their explorations of how sports teams function. For example, Eccles and Tanenbaum look at several aspects of coordination in team sports [4]. Eccles presents a framework that identifies communication, assumptions, planning, and reflection as critical elements in shared activity [4].

Studies of coordination in multiplayer digital games have also been performed. Researchers have examined coordination in team raids [14] or in larger-scale interaction patterns [2,24]. Others have considered coordination in real-time distributed games [14], and have focused on the awareness information that is necessary to support fast coordination of activity [16,17]. Other studies have looked at the coordination problems caused by network delays, showing that delays of even 100ms can have substantial effects on people’s ability to coordinate tightly-coupled actions [8].

A few researchers have identified that designing for coordination can be a problem in games. Tatar et al. [25] indicate shortcomings in current methods for designing games, and begin to look at design ideas for coordination in playground games. They investigate coordination in simple games based on rules, roles and turn taking, and explain coordination in playground games based on three factors: plurality, appropriability and aecompetition.

Games, like sports, are an interesting case for coordination because the difficulties of shared work can actually be an important part of the fun of the activity – and researchers have looked at the explicit connections between coordination and sociality [14,25]. In many of the work domains studied in previous research, management of interdependencies is required to achieve productivity, but in multiplayer games, the process of play and its emotional impact are more important than the outcome. Collaborative scenarios are therefore artificially created to provide enjoyment rather than arising implicitly from the shared task. As a result, game designers must be able to balance the coordination requirements such that an activity is possible, difficult enough to be challenging, and fun for the group. However, tools and processes for designing or even discussing these collaborative game elements have been missing. In the next section, we address some of these problems through the PLATO framework.

3. PLATO: A GAME COORDINATION FRAMEWORK

The PLATO framework provides game designers with a tool for conceptualizing, describing, and analyzing specific coordination episodes in multi-player games. Coordination episodes are situations where players must coordinate to achieve a goal or to improve an aspect of their performance (e.g., completion time).

To specify the details of a coordination episode, the framework must identify relevant conceptual entities that can play a role in the coordination. Malone and Crowston’s theory [13] employs four entities, which are valuable but insufficient to capture the subtleties of coordination in digital games. We have adapted these ideas to better suit the specific needs of multi-player game design. In PLATO, we assume that the goal of the episode is determined by the game designer, and that specific types of interdependencies should be explicitly modeled in the framework. Therefore, we build PLATO on five concepts that can engage coordination requirements: Players, Locations, Actions, Time, and Objects.

3.1 PLATO Part 1: Players

A multi-player coordination episode depends on players carrying out actions in the shared virtual space of the game. Obviously, the number and type of players (or their characteristics) is a central component of the coordination episode. Players are part of every PLATO analysis, as they are the agents in the game world. There are two primary aspects to specifying a player element: which game entities can act as players, and what degree of specificity is needed to characterize the coordination episode.

What game entities can act as a player?

Players are generally avatars or characters that can be directly controlled by people – for example, the player is obvious in first-person games (e.g., Enemy Territory) or third-person games (e.g., Tomb Raider or Super Mario Bros.) where the human directly controls a single avatar explicitly represented in the game world. Variations on this general case exist, however. In some cases the player is represented by a vehicle (e.g., a car in a racing game),
and in strategy games (e.g., Starcraft), the player can be the invisible but omniscient commander, or could be an individual unit in the game. This indicates that some games (e.g., team sport games or RPGs) have a number of playable avatars.

For our purposes, players can be any of the avatars or units that could be directly controlled by the human player, including those temporarily under the control of the game’s AI. Our definition uses an earlier organization of player types [15] in three tiers: main characters which are under player control, characters who assist the main character as henchmen or assistants, and characters who passively join parties to provide limited aid and advice.

How are players specified?
There are several ways that game requirements could constrain a player’s identity, role, or capabilities in a coordination episode.

- **Specific role or characteristic.** The player must be of a particular type (e.g., character class in an RPG), have certain attributes (e.g., a particular team) or have particular abilities (e.g., be able to defuse bombs).
- **Identity.** The episode can only be carried out by a specific character, or even a particular human-player identity. (e.g. only the initiator of a quest instance can participate in dialogue).
- **No constraint.** Any player can fulfill the requirement.

3.2 PLATO Part 2: Locations

Locations are points or areas in the game space where players can be, where coordinated activities can occur, and where objects can be placed. Most game worlds have a spatial organization of some kind (e.g., a sports field in FIFA Soccer, rooms and passageways in Portal 2, or a world metaphor in World of Warcraft), and this reference frame is used to specify locations when describing coordination episodes with PLATO.

- **Points vs Regions:** Locations are points in space corresponding the precise position of a player or object. Locations can also be labeled regions, which can be static like a room or area, or dynamic like the cone of vision of a guard in a stealth game.
- **Absolute vs. relative.** Locations can be defined based on absolute coordinates of the underlying game ‘map’ (e.g., a particular room where resources are stored, or a home region for a capture-the-flag game). Locations can also be relative (e.g., jumping onto a vehicle requires that the player is nearby, regardless of the map). Another type of dynamic relative location is seen in the ‘zones of control’ used in strategy games.
- **Specificity.** Different types of locations can be specified, including specific points, distances, or areas.

3.3 PLATO Part 3: Actions

An action is any change made to the game environment by a player – e.g., kicking a ball in a soccer game, or casting a spell in WoW. Actions are involved in most coordination episodes, although it is possible to construct episodes without them (e.g., a territory-based game might require that three different players simply exist in three particular locations simultaneously, without carrying out any particular actions at those locations). Actions are, of course, dependent on the nature and domain of the game.

In some cases, activities that are closely related to other PLATO elements (particularly locations and objects) are not represented in descriptions of the episode. For example, “move to location X” or “hold object Y” are usually implied by constraints in the location or object elements. Similarly, actions can imply an object, and in these cases it is often not necessary to explicitly state the object as part of the coordination. For example, an action such as “fire gun” implies that the player has a particular object (i.e., a gun), and the object does not need to be stated. These flexibilities in scenario specification underline the fact that PLATO is not intended as a formal specification tool, but rather as a resource for designers to discuss, understand, and plan coordination requirements.

3.4 PLATO Part 4: Time

Time is a critical aspect of coordination in multi-player games, and temporal constraints on a shared activity are often the key component used to make the activity challenging. There are two types of time requirements in PLATO: ordering and clocking.

- **Ordering.** Some activities must be carried out in a particular order (e.g., in Enemy Territory, the bridge must be built before the tank can be moved forward). One sub-type of ordering is **simultaneity,** where actions must occur at the same time.
- **Clocking.** Some activities must happen at particular times, or during particular time periods, or before a particular time limit. Two sub-types of clocked coordination are **soft clocking,** where there is some leeway on the timing of the activity (e.g., within ten seconds of time T), or **hard clocking,** where the temporal requirements are strict.

From these two main concepts, we define several specific types of temporal coordination constraints that are used in multi-player games. We note that this is a main area in which PLATO goes beyond previous coordination frameworks: where two temporal classifications were sufficient in Malone and Crowston’s framework, we define numerous types that better capture the richness of process interaction inherent in games. In the list below, we use the term ‘operation’ to imply either an action by a player, or an event such as a player occupying a location.

- **Mutual Order.** A series of operations must be performed in a specific sequence.
- **Clocked Order.** Operations must be completed by each actor before a timer expires.
- **Individual Order.** A single player must perform a series of operations to allow other players to proceed.
- **Soft-Simultaneity.** Several operations must be completed within a short time period (Tss).
- **Hard-Simultaneity.** A series of operations must be accomplished with negligible delay (Ts).
- **Parallel Activity.** A final contingent objective is achieved by players completing a series of operations independently.
- **Clocked Parallel.** Parallel activity with a time limit.
- **Order-in-soft-simultaneity.** A specific ordered set of interdependent operations, performed in a limited time.
- **Bound.** An operation by one agent must start prior to and end after another player’s operation.

3.5 PLATO Part 5: Objects

The final element in PLATO is the Object – a non-playable game element that is required for the coordination episode. PLATO considers three kinds of objects: passive objects which only have a state, and no actions (e.g., a key for opening a locked door or a flag in capture the flag; active objects, which have a limited set of contingent actions (e.g. a mine in a multiplayer shooter can explode when an enemy is proximate); and third, AI objects, which are capable of several actions. Unlike playable characters under AI control in a sports game, AI objects cannot be controlled by the player. Most enemy units would be AI-active objects.
The distinction between Player and Object can be subtle, and can change during a single scenario, but these subtleties can be dealt with by the designer. While we feel that our definitions are the most widely appropriate, designers can use alternate definitions suited to their game. Changes to definitions of Players and Objects do not affect the overall methodology or utility of the framework.

4. ANALYZING COORDINATION

PLATO provides a set of tools to assist game developers and researchers by providing a common language and criteria for formulating and quantifying coordination. The first step in analyzing coordination using PLATO is to constrain the scope of analysis to a single coordination scenario. Games are complex, and will likely incorporate many intermediate activities and goals. If a discrete game goal contains one or more coordinated actions or events then it is a coordination scenario, and can be analyzed using PLATO and associated notation.

In Eccles' work [4], atomic actions are the elementary actions achieved by a differential division of labor. Atomic analysis in PLATO is based on time. Analysis begins with the assignment of a time tag to a set of activities (for example T1–T2), which implies the time range from T1 to T2, | T1 ~ T2, see Table 1 for details).

Table 1. Time elements and their diagram notation

<table>
<thead>
<tr>
<th>Timing</th>
<th>Symbol</th>
<th>Sample Diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mutual Order</td>
<td>![Mutual Order Symbol]</td>
<td>![Mutual Order Sample Diagram]</td>
</tr>
<tr>
<td>Clocked Order</td>
<td>![Clocked Order Symbol]</td>
<td>![Clocked Order Sample Diagram]</td>
</tr>
<tr>
<td>Parallel Activity</td>
<td>![Parallel Activity Symbol]</td>
<td>![Parallel Activity Sample Diagram]</td>
</tr>
<tr>
<td>Clocked Parallel</td>
<td>![Clocked Parallel Symbol]</td>
<td>![Clocked Parallel Sample Diagram]</td>
</tr>
<tr>
<td>Individual Order</td>
<td>![Individual Order Symbol]</td>
<td>![Individual Order Sample Diagram]</td>
</tr>
<tr>
<td>Simultaneity</td>
<td>![Simultaneity Symbol]</td>
<td>![Simultaneity Sample Diagram]</td>
</tr>
<tr>
<td>Soft-Simultaneity</td>
<td>![Soft-Simultaneity Symbol]</td>
<td>![Soft-Simultaneity Sample Diagram]</td>
</tr>
<tr>
<td>Bound</td>
<td>![Bound Symbol]</td>
<td>![Bound Sample Diagram]</td>
</tr>
<tr>
<td>Order in Soft-Simultaneity</td>
<td>![Order in Soft-Simultaneity Symbol]</td>
<td>![Order in Soft-Simultaneity Sample Diagram]</td>
</tr>
</tbody>
</table>

PLATO has utility as a set of concepts for analyzing and discussing coordination in collaborative games, but to increase the power of the framework in more practical scenarios, we have also devised a diagram notation to help designers visualize how the different coordination elements interact in a given collaborative scenario. We propose two types of charts, Player-Action-Time (PAT) and Location-Player-Time (LPT). Other types are possible, but we have found these to be the most descriptive. In PAT charts, time is on the x-axis and players on the y-axis. Annotated actions for each player are drawn as horizontal lines, and timing and collaborative relationships are drawn as a second set of lines according to the notation guide in Table 1. PAT charts are particularly useful for exposing parallelism and simultaneity in collaborative scenarios. LPT charts also have time on the x-axis, but have labeled location on the y-axis and plot players as lines moving from place to place over time. LPT charts aid understanding of how space is used in collaborative scenarios.

To demonstrate how these graphs are created, we examine a simple scenario in which two players must move through a dark hallway with spring-loaded switches at either end. One player must hold the first switch while the second traverses the hallway, then the second player holds the far switch while the first player moves, (Figure 1). The location diagram clearly shows the transient-residency pattern imposed by the collaboration in the sloped and straight lines for Players A and B. The timing diagram shows the mutual order actions (Bound) associated with the holding of a switch and hallway movement.

Figure 1. PAT (top) and LPT diagrams for the dark hallway

Table 2. Analysis of a scenario in the game Commandos.

<table>
<thead>
<tr>
<th>Elements</th>
<th>Entities in the game</th>
</tr>
</thead>
<tbody>
<tr>
<td>Players</td>
<td>Green Beret, Marine, Driver</td>
</tr>
<tr>
<td>Locations</td>
<td>L1, L2, L3, L4, The areas under the vision range of soldiers</td>
</tr>
<tr>
<td>Actions</td>
<td>throwing cigarette, killing the soldier, carrying the dead body</td>
</tr>
<tr>
<td>Times</td>
<td>mutual order-in-soft-simultaneity</td>
</tr>
<tr>
<td>Objects</td>
<td>Cigarette</td>
</tr>
</tbody>
</table>

A sub-scenario of Commandos (Figure 2) contains a complete set of coordination factors. Players must perform a number of coordinated actions to break into a sentry point. The players are composed of Driver, who carries a cigarette, Marine, equipped with a harpoon gun, and Green Beret. To complete their objective, Driver should throw the cigarette in front of one enemy soldier. Almost simultaneously Marine should kill the second soldier with his harpoon and shortly thereafter Green Beret should carry the dead body away. Timing in this scenario is order-in-soft-
simultaneity. The cigarette is a coordination object because its appearance sets the collaborative task in motion.

![Diagram of Commandos scenario with annotations](image)

**Figure 2.** Coordination episode in Commandos, showing locations of soldiers (L_1, L_2), their views (triangles), cigarette (L_3) and commandos (L_4) at the beginning of the scenario.

This scenario starts at (0-T_1): when Driver throws the cigarette (PAL), At (T_1-T_2) Marine shoots the target at the L_4 (PAL). At (T_2-T_3) Green Beret moves the dead body out sight.

PLATO provides an analytic framework for coordination activities in digital games, but it also serves as an analytic framework deployable throughout the design process. Two situations are provided to illustrate how PLATO can be used in design.

The first scenario, depicted in Figure 5, is imagined to take place during play testing. In this scenario the duration of the cigarette burn, and therefore the amount of time the first soldier is distracted, has been reduced. Green Beret dies repeatedly because he is caught by the first soldier while carrying the body of the second. This playability error would appear as an increased number of deaths in a traditional heat-map analysis, showing the undesirable result, but not the ultimate cause. In the LPT coordination diagram (Fig. 5), the return of soldier 1 to location 3 before Marine leaves is apparent, as is the expiration of the importance of the active coordination object (the cigarette).

Our second scenario is imagined to take place during a paper-prototyping design stage. In this case, Marine is able to both harpoon the guards and carry them away, and here, the scenario collapses because Green Beret no longer has a role. Marine can simply kill both guards and drag them away with the help of the cigarette. If the level was played by two Marines, little coordination would be required.

The collaborative paucity of this scenario stands out clearly in the Player-Time coordination diagram. Once the cigarette is thrown, Marine kills and drags away both guards. Sketching diagrams like this one at an early stage can help design teams decipher if their collaborative scenarios have the desired degree of interdependency before the digital version of the game is built.

![Diagram of Commandos scenario with annotations](image)

**Figure 5.** Timing error collaboration design. The thick black line represents the return of the first soldier.

![Diagram of Commandos scenario with annotations](image)

**Figure 6.** The Commandos scenario when Marine has the ability to both harpoon and carry dead bodies.

### 5. CASE STUDY: PORTAL 2 CO-OP

Although multiplayer games exist across genres, the unique characteristics of Portal 2 co-op mode – particularly the deliberate and explicit pacing of collaborative acts – makes it an excellent choice to illustrate PLATO analysis. Portal 2 is a first-person puzzle-platform developed and published by Valve Corporation. The game’s robotic characters, Atlas and P-Body, are equipped with weapons which create portals – wormhole-like passageways which link two spaces – to solve challenges posed by a maniacal AI obsessed with quality assurance. In Portal 2, co-op mode consists of five chapters featuring individual test chambers which must be traversed without being destroyed and feature puzzles which are designed to be impossible to complete individually.

#### 5.1 Experimental Setup

Two players played the co-op mode for approximately 8 hours to learn the game mechanics prior to analysis. Both players returned to the beginning of the co-op campaign to replay the levels. As the puzzles had been solved once in the first run, the second run focused on using PLATO to analyze the coordination tasks. During the second run, 20 scenarios were completed by players playing in separate rooms, while screen capture recorded their progress. Three illustrative scenarios are described here.

Each chamber was analyzed using video annotation in two phases. In the first phase all collaborative elements in the scenario were identified and labeled. In the second phase, the timing and...
collaborative behaviors of the collaborative tasks were charted. As PLATO focuses solely on what players did rather than why they did it, no speak-aloud, retrospective or otherwise was required. Details on the world and gameplay of Portal can be found in [19].

5.2 Results
In this section we present three scenarios encoded (Chapter, Level, Chamber). Coordination elements are charted to demonstrate the scope and utility of the PLATO framework.

5.4.1 Scenario 1 (1,1,1)
Players start the scenario (1,1,1) (Fig. 7, Table 3) in the first room of a series of three connected partitions and proceed to the exit in the third room. The analysis encapsulates two sub-scenarios.

The first sub-scenario is entering from the second room from the first. At (0 - T2): Player A should place a portal at a valid location (P1) in the first room (L1) (PAL). At (T1 - T2) Player B must stand on the weighted switch in the first room (WS1) which opens the first door (PAL). At (T2 - T3) Player A enters to the second room (L2) (PL). At (T3 - T4) Player B stands on the second weighted switch (WS2) in the first room which opens the second door (PAL). At (T3 - T4) Player A must move to the third room (L3) (PL). At (T4 - T5) Player A should place their second portal in the third room (P2) (PAL). (T5 - T6): Player B should use the portals (P1→P2) to move to the third room (PL).

5.4.2 Scenario 2 (1,2,2)
In (1, 2, 2), players must perform similar tasks using soft-simultaneous actions by pushing a series of buttons in rapid sequence to release an Edgeless Cube. The room (shown in Figure 10) is completely symmetric with two buttons (S) on each side but at different heights. Each player must press two buttons in a short amount of time, requiring portals to cut the distance between two buttons; this implies order in soft-simultaneity (T^SS1 and T^SS2) specifically, and soft-simultanecity generally. Table 4 illustrates the component analysis of this scenario. At (0 - T1) both players create a portal set which connects their individual button locations (P1, P2, and P3, P4) (PLA). At T1=T^SS (T^SSI) Players push their first buttons (PLA). At T^SS2 players push their second buttons (PLA). Finally, either of the players can unlock the final door.
The timing diagram in Figure 11 demonstrates the independent action allowed prior to the first button press, then the rapid simultaneous action required for the second press. The final step of unlocking the door is not shown as it is not collaborative.

<table>
<thead>
<tr>
<th>Time</th>
<th>Object</th>
<th>Action</th>
<th>Entities in the game</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>P</td>
<td>Button</td>
<td>Location Placing Cube’s Location</td>
</tr>
<tr>
<td>T2</td>
<td>P</td>
<td>Button</td>
<td>Location Placing Cube’s Location</td>
</tr>
<tr>
<td>T3</td>
<td>P</td>
<td>Button</td>
<td>Location Placing Cube’s Location</td>
</tr>
<tr>
<td>T4</td>
<td>P</td>
<td>Button</td>
<td>Location Placing Cube’s Location</td>
</tr>
<tr>
<td>T5</td>
<td>P</td>
<td>Button</td>
<td>Location Placing Cube’s Location</td>
</tr>
<tr>
<td>T6</td>
<td>P</td>
<td>Button</td>
<td>Location Placing Cube’s Location</td>
</tr>
<tr>
<td>T7</td>
<td>P</td>
<td>Button</td>
<td>Location Placing Cube’s Location</td>
</tr>
<tr>
<td>T8</td>
<td>P</td>
<td>Button</td>
<td>Location Placing Cube’s Location</td>
</tr>
</tbody>
</table>

### 5.4.3 Scenario 3 (1,3,2)

In this seemingly complex scenario (Figure 12), the players must collaborate to re-purpose lasers to both destroy turrets and unlock the final door; which comprises the two sub-scenarios in the test chamber. The analysis of scenario (1,3,2) is shown in Table 5.

<table>
<thead>
<tr>
<th>Elements</th>
<th>Entities in the game</th>
</tr>
</thead>
<tbody>
<tr>
<td>Player</td>
<td>P-Body, Atlas</td>
</tr>
<tr>
<td>Location</td>
<td>Laser sensitive area, accessible part of laser beam, location of the cube, the area of the exit door, location of portals, location of the button</td>
</tr>
<tr>
<td>Action</td>
<td>Creating Portal, Changing the cube’s position and direction, pushing button</td>
</tr>
<tr>
<td>Time</td>
<td>Individual ((T_0 - T_4)), Bound ((T_4 - T_6)), Parallel ((T_6 - T_8)), Order ((T_8 - T_9))</td>
</tr>
<tr>
<td>Object</td>
<td>Redirection Cube</td>
</tr>
</tbody>
</table>

In the first sub-scenario, players must obtain a redirection cube to aim an environmental laser at a series of turrets without being destroyed by either the laser or the turrets. At \((0 - T_1)\) player A should push a button \((S)\) to release a Redirection Cube \((PA)\). At \((T_1 - T_2)\) player A should move to the location of the cube \((PL)\). At \((T_2 - T_4)\) player A places the cube under the thermal discouragement beam \((PALO)\). At \((T_4 - T_5)\) player A should change the direction of the cube to aim the beam towards the turrets \((PALO)\). Over \((T_3 - T_6)\) player B should stand on a weighed switch \((WS)\) allow player A to see the turrets \((PAL)\). The only collaborative act in this sub-scenario is the division of labor, where one player must open the window to expose the turrets while the other destroys them with the laser, illustrating the Bound constraint. Player B must stand on the switch before Player A starts destroying turrets, and must leave the switch after the turrets are destroyed, but is otherwise unconstrained in time.

The second scenario involves directing the laser through a series of portals to a laser-powered lock opening the final door. At \((T_6 - T_7)\) players should place two portal sets \((P_1, P_2\) and \(P_3, P_4)\) \((PAL)\). At \((T_7 - T_9)\) either player should aim the laser beam through the portals toward the lock using the cube \((PALO)\). The primary collaboration in this instance is coordinating the placement of the portals, which is time independent and tied to relative position. What appears to be a complex scenario, involving multiple objects, switches and actions, devolves to a relatively simple coordination scenario in the diagram shown in Figure 13. Player B is irrelevant until T3, then must stand on a switch while Player A completes the action. The subsequent parallel portal creation must be performed correctly but is limited in temporal complexity.

### 6. DISCUSSION AND FUTURE WORK

The case study demonstrates the value of the PLATO framework for describing, analyzing, and re-designing coordination scenarios. The framework provides a cogent and consistent representation of aspects of game collaboration that have previously been difficult for designers to conceptualize, discuss, and evaluate. PLATO builds on previous frameworks by Malone and Crowston [13] and Eccles [4], but adapts the structure of the framework to reflect the differences between workplace and digital-game coordination. The perspective of the framework is the realization that whereas workplace coordination is grounded in the requirement to complete a task (with measures of success related to the quality, speed, and efficiency of task completion), in games the process is the outcome, and therefore a more nuanced description of the elements of coordination is required. Temporal requirements in particular can have many different instantiations because of the real-time nature of the activity, and because the timing constraints in games can be used to change the experience of the scenario.

The design examples demonstrate PLATO’s capacity to provide practical insight during game design and testing, by providing a common language, a diagramming nomenclature, and a structured analysis technique. The goal of the PLATO framework is utility for designers, so the diagrams and descriptions built from the framework are descriptive rather than prescriptive, and, there is considerable flexibility in the framework for designers to contextualize the idea to fit the particular domain of their game. For example, there is space for designers to adjust the distinction between player and object, in a manner that is consistent with their game and genre. Given the complexity of digital games, there are many grey areas in any classification scheme. With PLATO we do
not provide a rigid decomposition, but rather a tool for describing and analyzing multiplayer interaction.

As a descriptive solution, PLATO is only one of many possible techniques. We provide what we believe to be a compelling and useful general framework for the analysis of coordination in digital games. However, other general descriptions are possible, and extensions and refinements of PLATO could be useful for specific genres of games or classes of interaction. For example, different refinements of the framework could be developed for the more loosely-coupled coordination and asynchronously-evolving gameplay in FarmVille or MafiaWars, and another for the tight real-time gameplay of an online shooter.

We see several directions for further work with PLATO. First, we will test the utility of the framework in studies with game designers. Second, we will characterize multiplayer interactions from different domains, including types of coupling in coordination. Third, we will determine whether the framework scales well to larger coordinated episodes with more players. Fourth, because the framework specifies only the coordination requirements of an episode, and not the ways that players might satisfy those requirements, we will add guidelines for managing awareness information [1,6,17].

7. CONCLUSION

Coordination is a critical element of many multi-player digital games, but it is currently difficult to describe and analyze coordination episodes in games. In this paper we presented PLATO, a framework for the analysis of multiplayer coordination in digital games. This work extends existing research characterizing the role of coordination in workspaces [13] to capture the more nuanced role that coordination and timing plays in digital games. The framework provides a set of concepts, an analysis process, and a diagram notation, and can be used by designers to understand, evaluate, and manipulate coordination episodes both in early designs and during playtesting. PLATO provides the first comprehensive support for improving the design of a critical element of a wide variety of multiplayer digital games.

8. ACKNOWLEDGMENTS

We would like to thank the GRAND-NCE for funding.

9. REFERENCES