# SCCRASL and CADGE: Crisis Representation and Simulation in Serious Games

Terry Bossomaier, James Tulip, John Carroll, David Cameron
Centre for Research in Complex Systems
Charles Sturt University
Bathurst NSW2795, Australia
Email: tbossomaier@csu.edu.au

## Abstract

Good communication is essential to successful aversion or management of crises, but it raises a number of challenging issues. These vary from the technical, the provision of the right information resources at the right time, to the sociological, the finessing of differing priorities and agendas amongst stakeholders. Communication between stakeholders may be made more difficult by different cultures, ontologies and silo attitudes. This project forms part of a new approach to crisis communication planning and training which embodies serious games and applied drama. The paper describes the integration of a scripting language, CCRASL, with simulation of crisis scenarios in the development of this new genre of serious game.

Keywords: serious games, DEVS, simulation, applied drama

## 1 Introduction

As global warming, rising sea levels, food shortages and disease pandemics confront us every day in the news, the need for better techniques of handling crises does not need over much justification. Computer games, referred to as serious or epistemic games (Carroll *et al.*, 2006; Anderson *et al.*, 2009), when used for non-entertainment, educational or strategical goals, have featured in disaster planning for a number of years (see popular articles such as (Christopher, 2005)). The scale of these games varies enormously. At the one extreme we have military simulations involving perhaps thousands of people (Lee *et al.*, 2009). At the other we have games for use by individuals or small numbers of people. For example, Wickler et al. (2007) describe *I-sim*, a tool for integrating scenarios for emergency management training into an agent framework. Instructors can modify scenarios on the fly, with a focus on planning, system tools and logistics. This project is inclined towards the latter smaller scale end, and thus avoids more heavyweight integration such as the High Level Architecture (IEEE Standard 1516).

However, effective handling of large scale crises requires good communications and authors such as Coombs (2007) and Heath and Miller (2004) consider this aspect in some detail, while authors such as Goh et al (2006) address the interface between media and real-life crisis management through the construction of crisis communication portals strongly emphasising mobile phones.

The research in this paper forms part of a large project which explores the new area of the integration of applied drama and serious games to improve crisis communications. Scenarios are generated by domain experts (e.g. emergency services) and expressed in a purpose designed language. The program so written is then compiled to produce output for a game engine, the **C**ommunication **A**pplied **D**rama **G**ame **E**ngine (CADGE) which runs the serious game. In the present case CADGE takes an XML input which is generated by the the CCRASL compiler output.

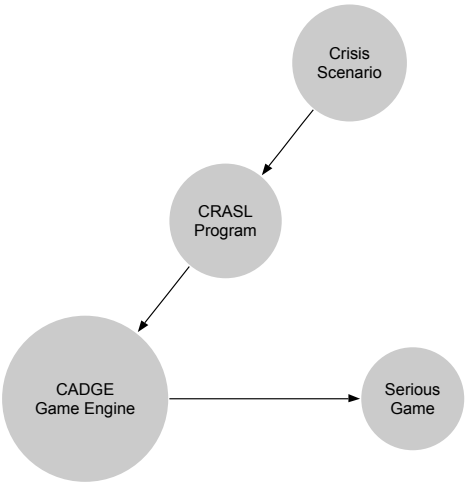


Figure 1: The Crisis Communications Scenario Pipeline

Figure 1 shows the production pipeline.

There is an important distinction here between simulation and games. The serious games created for crisis communication are for communications personnel. Alongside these games scenarios will evolve involving other stakeholders and world events. At one extreme the entire action may be involve only the flow of information and mis-information, as, say, in a political scandal. At the other extreme a large scale environmental event may be taking place, and the communications game has to take place against a clock in which a simulation of this event is running. An example might be the simulation of the spread of bushfire, the firefighting resources, the evacuation plans, all combined in a sophisticated agent-based model of what is happening. The public relations decisions will depend on the current state of the simulation and will feed back into the simulation itself.

Section 1.1 discusses the applied drama approach and section 1.2 introduces the need for a language, CCRASL (**C**risis **C**ommunication **R**epresentation **A**nd **S**imulation Language, to generate a family of serious games within a domain, and section 1.3, makes the link to simulation methodologies.

## 1.1 Applied Drama

The use of role playing for exploring scenarios and training has a long history ranging across military war games through to the corporate board room (Cameron & Carroll, 2009; Carroll *et al.*, 2006; Anderson *et al.*, 2009). This project breaks new ground in integrating applied drama with human players and a *drama master,* DM, with computer games generating scenarios and releasing media resources to players.

An essential part of the applied drama methodology is the *chat* system built into CADGE. This enables the DM to interact with the players and to assess the level of confidence and tension, making adjustments where necessary. However, this is a general feature which does not need to be specified in the CCRASL script. But the groups are useful in determining the target audience of any chat messages or broadcasts.

## 1.2 CCRASL

Creating computer games is expensive. It takes a lot of time from teams of experts, comprising game designers, interface and graphics specialists and computer programmers. Thus many game franchises have their own scripting languages with which modifications of a game may be made. However, for serious games something more is needed, in which diverse families of games can be built by people who are experts in their application domain (e.g. bushfire control), and do not have the game development skills alluded to above.

We want endusers to be able to write games in a language with which they can easily work. To this end the language needs to

- use self-evident syntax rather than the arcane syntax in low level languages such as C. We need to have easily understood full English keywords and a quasi natural language style.
- be compact. As much as possible must be hived off to implicit libraries
- but must use as many generic library functions (rather than game specific) as possibe

- *integrate smoothly with simulations*

XML has become the *lingua franca* for the web and it serves well as an *intermediate* language for glueing components together. But it is not suitable as the game description language for a variety of reasons:

- XML gets very verbose for complicated systems;
- XML is weak on specifying control;
- The RDF and Schema systems would not allow some of the compile time checks we could introduce. For example determining if an action with a wild card pattern could use all the actions speficied would be complicated (if possible at all) in XML.

CCRASL intends to meet these requirements. This paper describes the language components and the simulation integration, but the detailed syntax is considered elsewhere.

We can compile CCRASL to different intermediaries. In this paper we compile to XML (generating verbose output for a computer to read is fine). But we could also compile to simulation frameworks such as Zeigler's DEVS (Zeigler *et al.*, 2000), described in section 1.3. But heterogeneous destinations are also possible. Imagine we had to hand a game about a tsunami emergency. One might compile to the scripting language of the game with an additional public relations layer added.

In second generation of CCRASL, referred to as SCCRASL, the game becomes an atomic model in DEVS and external simulations are added as a coupled model, defined in section 1.3 and eqns 1 and 2.

### 1.3 Simulation using DEVS

There are numerous simulation methodologies in existence, but one which fits particularly smoothly with the CCRASL approach is Zeigler's 196 model DEVS (Zeigler *et al.*, 2000).

The DEVS atomic, deterministic model uses 7-tuples of the form

$$M =< X, Y, S, ta, \delta_{ext}, \delta_{int}, \lambda > \quad (1)$$

| | | |
|---|---|---|
| | X | is the set of input events |
| | Y | is the set of output events |
| | S | is the set of states of the system |
| where | lt | defines the lifetime of state S |
| | $\delta_{ext}$ | is the *external transition funnction* |
| | $\delta_{int}$ | is the *internal transition function* |
| | $\lambda(S)$ | is defines output at end of life of S |

The power of DEVS, however, lies in its hierarchical structure, allowing *coupled models* to be built from atomic and other coupled models. A coupled model $N$ is an 8-tuple and takes the form

$$N =< X, Y, D, M_i, C_{xx}, C_{yx}, C_{yy}, Select > \quad (2)$$

$X, Y$ are input and output events (to the components) as before. $D$ is the nameset of the component models (which may be atomic, or, themselves, coupled models) and $M_i$ is the set of these models. The $C$ terms couple the models together: $C_{xx}$ couples the inputs to the model, $C_{yy}$ couples the outputs from individual models and $C_{yx}$ handles the internal couplings where output from one model becomes the input to another. Finally *Select* is a function to resolve conflicts arising from simultaneous events.

There are very many DEVS applications and refinements now extant. Relevant examples include Vangheluwe and Vansteenkiste's description of cellular automata (CA) in DEVS (Vangheluwe & Vansteenkiste, n.d.) and Müller's *Mimosa* an agent based modelling package (Müller, n.d.). CAs are in widespread use for modelling many physical processes and would be a natural choice for the flood scenario described below. Agent based models add additional simuation elements of human (cognitive) entities.

## 2 CCRASL Structure

Unlike traditional programming or scripting languages such as Java or Python, CCRASL is built round the idea of *state transitions* rather than sequential control flow. A transition has something in com-

mon with the tuples used in DEVS, but is specialised for the crisis communication world. Transitions combine together the building blocks of the game, devices, media resources and world events.

Because CCRASL is built for communication applications it is sensible to include some input/output operations and messaging primitives within the language In many other respects CCRASL relies on libraries or even entire programs in order to maintain a high level of abstraction and render it useable by the non-programmer.

Input is straightforward. But there is a novel feature in output, where the *dashboard* concept is embedded within the language itself. It is useful to do this because so many of the events which occur in a game are precisely about media resources and to be able to monitor them in real time is essential.

## 2.1 Components of a CCRASL Program

Bearing in mind that the target audience is computer non-specialist, the CCRASL specification uses English liberally, with reserved words, is sparse on symbols and uses the natural structuring style of text documents. Thus a program consists of a series of parts, each beginning with a heading. They are written in upper case followed by a period, and must occur in sequence, all must be present (although could be empty) and comprise: CCRASL GLOBAL PRE-TEXT (§2.1.1; CCRASL DEVICES (§2.1.2); CCRASL PERSONAS (§2.1.3; CCRASL CHARACTERS (§2.1.4; CCRASL GROUPS §2.1.5); CCRASL STATES 2.2 and CCRASL TRANSITIONS (§2.2.1). Note that the reserved word CCRASL precedes each and must be separated from the next word by a minimum of one followed by zero or more white space characters. The reason for the prefix is to enable common words (such as character) to appear in text sections, such as the global pre-text without explicit escape sequences. Since CCRASL refers to the language it is highly unlikely it would ever be used in text about the game.

### 2.1.1 Global Pre-text

The CCRASL GLOBAL PRE-TEXT is free form text describing the setting of the game (§3.1), terminated by the next part-heading. It can contain arbitrary tags, but no keywords.

### 2.1.2 Devices

The central activity of the game is the transfer of messages among players and entities (agents) and at first sight it is not clear why one should need a device category. But in fact this adds considerably to the scenarios which can arise. Firstly messages now go to devices and thus may be shared amongst more than one agent, intentionally or otherwise. Secondly devices allow format and message length restrictions and can be disabled in some circumstances (e.g. crash of the mobile phone network during the London bombings). Devices support media types, given as reserved words: **audio; video; text; image;** and **document** (which may contain text and images). Devices may belong to groups (§2.1.5) and form part of the state space, $S$ in eqn 1.

### 2.1.3 Personas

Characters have personas and each persona has rights and priorities associated with it defining what things in the game it can modify. The drama master has absolute control. Personas have a priority, 0–10. The DM would have 10, analogous to *root* in UNIX systems. The priority determines which character will get to act when several have a matching pattern. Personas also have a set of devices to which they can send and from which they can receive. The persona will normally have a descriptive pre-text, so that a player assigned such a persona can become familiar with its mindset.

### 2.1.4 Characters

Characters are individual agents within the game. In the examples so far they are human players but may also be AIs within the game. Each character has a persona from which it inherits rights and devices (analogous to the class/object software relationship). Con-

tinuting the UNIX metaphor, characters may belong to groups (§2.1.5).

The players generate the input events, $X$. The output events, $Y$ are non-null only when the game ends and are generated by the game termination states. In coupled models, where the game is combined with external simulations, additional input/output events will go between the models.

#### 2.1.5 Groups

Both characters and devices belong may belong to one or more groups. A message may be sent to a group of devices and thus be accessible to a range of characters.

### 2.2 States

States lie at the heart of the program and describe characteristics of the environment and the devices, i.e. the states $S$ in eqn 1 form a product space, comprising the states of devices, the world and the game. Media events have different time elapse functions. For example, a fax will stay on a fax machine until it is actively removed. A news broadcast occupies a particular time slot and so on.

In the simplest case there are no world states, but just the set of device states and the game active or termination states. There is no global clock running.

SCCRASL introduces a simulation clock (which can still on occasion be slowed or accelerated by the DM). The simulation models, $i$, may have many states, but only some of these might be communicated back to the game model, $g$, via the couplings $C_{ig}$.

#### 2.2.1 Transitions

Finally after the definition of all the entities within the game the transitions are described. Most of them are device transitions, in which a media resource is released to a device. The resource may become available immediately in full, such as an email, and remain in situ until removed (such as the deletion of an email). Device transitions usually result from inputs $X$ from the players, including the DM.

Transitions may also occur in the world states (e.g. dam bursts in the flood scenario, section 3) and these in turn may trigger media releases to devices.

The states in the DEVS model are simply the states of the devices (whether they have accessible media etc).

### 2.3 Dashboard

The dashboard provides charts of various kinds of what is going on. It includes all the conventional displays, graph, pie, histograms. It also needs access rights (e.g. the DM would be able to see everything, but players would have more restricted views). It provides access to all the devices a player can see (and no more). Different game engines could implement this in different ways and the language does not control any display related issues, such as screen layout, foreground/background/hidden information and so on.

### 2.4 SCCRASL: DEVS integration

In SCCRASL, version 2 of CCRASL, the game component becomes a model in a DEVS coupled model. Some of the transitions (those of the human players) remain the same. But others are generated by the other (simulation) models, through the couplings, $C$. So, in The Flood example in section 3, the leakage of water from the dam would be a simulation which will run autonomously. But it will generate output events (such as volume of water leaked, visibility of leakage etc) which become triggers for the release of media resources in the game component.

The coupled DEVS structure means that these simulation models do not have to be generated anew for each game. They are not themselves part of CADGE and can be sourced from elsewhere. For example, lava flow CA models, such as Barca et al (1994) could be used in modelling the crisis unfolding from a potential volcanic eruption.

## 3 A Simple Example: The Flood

In organisational and public affairs terms, a crisis is an event, revelation, allegation or set of circumstances which threatens the integrity, reputation or

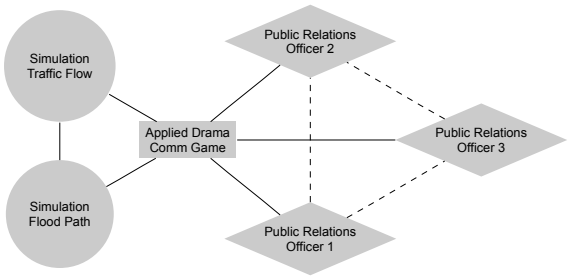


Figure 2: A coupled model for The Flood. DEVS components are shown as circles with the game model, a rectangle. The diamonds show the players in the game. Dotted lines show interactions between players using the chat system.

survival of an individual or organization (Galloway & Kwansah, 2005). The Flood is a public relations scenario relating to how people would react if the dam next to which they live were to burst. As with many emergencies of this kind, people would like to leave evacuation and abandonment of their homes to the last possible minute. As a consequence, roads may get blocked and evacuation made much more difficult. Politically the consequences of acting too soon and unnecessarily can be as damaging as not taking action early enough. Good communication is essential.

The game has only one persona, the public relations officer. Figure 2 shows a coupled model with three characters with the public relations persona.

### 3.1 The Pre-Text

In applied drama, it is the *pre-text* which initiates dramatic action and provides a firm base for the dramatic encounters to be played out in the scenario. In the game, the pre-text is a simple text statement which serves to define the nature and limits of the dramatic world and also to clearly imply roles for the participants. It also activates the expectations of participants and provides group coherence (Cameron & Carroll, 2009). Importantly, the pre-text positions the participant in a role rather than as an individual dramatic character (e.g. with a name, back story) in order to lessen the need for dramatic 'acting', or make believe on the part of the player. The pre-text for The Flood scenario tells the participants:

*You are a PR officer for the City Council. It has been raining for days in the area around Lagoon. For the first time since the dam wall was raised, water is overflowing into the new spillway for the Millenium Dam.*

A wide variety of resources are used by the game, listed in section 3.2. The pre-text is accompanied by the release of resources 1,2,3. The termination states are *crisis averted*, *panic* and *disaster*.

### 3.2 Flood Resources

The various resources used in The Flood scenario are described below. Most of the transitions between them occur after an elapsed time (not as a result of input events). This is a game time, and is not necessarily linear. The device transitions occur as a result of timing (such as the phone calls or videos) or as a result of input events (eg uploading of a template) Thus resources 14,18,9 and 11 are internal state transitions or result from actions of the DM. Resources such as 8 or 10 result from input events (player actions).

1. **Image** of dam spillway. Quotes from engineers, the Mayor, local farmers. Instructions to prepare press release from the Head of the Environmental, Planning and Building Services department.

2. **Newspaper** article - 8am *Dam filled at last*. After many months the dam is full, a new era of plentiful water for has begun, farsighted council has guaranteed the future of the region, congratulations all round.

3. **Phone call** - 8:05am Council general manager. Instructions to prepare press release re councils response to minor flooding downstream from new dam

4. **Phone call** 8:10am - the city engineer. There has been a report from the site engineer that there may be some issues with the dam. Mayor is anxious. Prepare a HOLDING STATEMENT. (a simple statement to the media that Council is responding, and that details will be forthcoming when they can be verified).

5. **Template** resource holding statement prepared and uploaded

6. **Local Radio** 8:30am - on the spot reporter claims dam wall is leaking. Fears that dam is structurally unsound are starting to gain momentum.

7. **Phone call** 8:31am Prepare PUBLIC AFFAIRS GUIDANCE (PAG, an internal briefing to remind Council officers on appropriate policy, rules and laws guiding Council responses to the crisis, and for handling media enquiries on the flood) and HOT ISSUES BRIEF (HIB, a document clearly spelling out the events so far, the nature of current and likely media enquiries, and Council responses).

8. PR officer prepares PAG and HIB for mayor and council staff.

9. **Phone Call** 9:30am Council General Manager where is PAG/HIB? There will be a media conference for mayor at 12pm. Prepare MEDIA ALERT, MEDIA RELEASE, TALKING POINTS (a brief for Council staff that suggests suitable responses to the questions the reporters are likely to ask, and statements that best reflect Council's position).

10. **Templates** Upload MEDIA ALERT, MEDIA RELEASE, TALKING POINTS

11. **Local Radio** 11:30am talk back radio caller suggests that all was not well in tendering and construction process: nepotism, bribes, political influence, faulty materials, slipshod methods.

12. **Local Radio** 11:32am Mayor's brother (interviewed at Airport) denies any irregularity in awarding of contract. Everything is fine. Goodbye.

13. **Template** Upload TALKING POINTS

14. **Video** footage, dam and spillway, the TV reporter describes the worst case scenario and questions Council's ability to act.

15. **Document** Engineer's report. Dam is structurally sound and despite some minor issues with the spillway gates it appears safe.

16. **Radio** Interviews with fearful residents preparing to leave homes.

17. **Document** Police and emergency services' evacuation plan unveiled.

18. **Video** 12:00pm - mayoral media conference WITH/WITHOUT updated talking points.

19. **Phone Call** 12:31pm Mayor's feedback on briefing WITH/WITHOUT updated talking points.

The release of these resources is caused by the actions of the players and the DM. In SCCRASL, the game becomes coupled (as a DEVS model) to one or more simulations, which unfold in real (game) time. The progress of the dam leakage and the subsequent flow of water can be modelled with cellular automata, for which there are plenty of examples. Another important consequence, if the crisis is not averted, is the evacuation problem and traffic flow consequences, also easily modelled with CAs.

# 4 Discussion and Further Work

The paper describes a language creating serious games centred on crisis communication. It enables a non-computing specialist to build a new game with different scenarios, personas and outcomes which will run directly in the CADGE engine. By incorporating compatibility with a standard simulation framework (DEVS) version 2 (SCCRASL) will allow the integration of arbitary simulations purpose written or reused from elsewhere.

Games such as The Flood run in CADGE after hand scripting the scenario and transitions in XML. The CCRASL compiler is under construction and the integration of Mimosa to provide agent based models using DEVS will be the subject of further work.

## Acknowledgements

This project was supported under Australian Research Council Linkage Project ARC LP0775418, *Crisis management simulation: developing a methodology for transforming communication response.*

## References

Anderson, M., Carroll, J., & Cameron, D. 2009. *Point of view: linking applied drama and digital games. Drama Education with Digital Technology*. Continuum International Publishing Group, London.

Barca, D., Srisci, G. M., Gregoria, S. Di, & Nicoletta, F. 1994. Cellular Automata for Simulating Lava Flows: A Method and Examples of the Etnean Eruptions. *Transport Theory and Statistical Physics*, **23**(1–3), 195–232.

Cameron, D., & Carroll, J. 2009. Lessons from applied drama: conventions to help serious games developers. *Pages 27–41 of:* Petrovic, O.and Brand, A. (ed), *Serious Games on the Move*. SpringerWien, NY.

Carroll, J., Anderson, M., & Cameron, D. 2006. *Real Players*. Continuum International Publishing Group, London.

Christopher, A. 2005. *Games tackle disaster training*. http://wired.com/news/technology/1,69580-0.html.

Coombs, W.T. 2007. *Ongoing Crisis Communication*. Sage.

Galloway, C., & Kwansah, A. 2005. *Public Relations Issues and Crisis Management*. Thomson/Social Science Press: Victoria.

Goh, O.S., Ardil, C., Fung, C.C., Wong, K.W., & Depickere, A. 2006. A crisis communication network based on embodied conversational agents with mobile services. *International Journal of Information Technology*, **3**(4), 257–266.

Heath, R.L., & Millar, D.P. (eds). 2004. *Responding to Crisis: A Rhetorical Approach to Crisis Communication*. Lawrence Erlbaum Associates, NJ.

Lee, Y-I., Trim, P., Upton, J., & Upton, D. 2009. Large Emergency-Response Exercises: Qualitative Characteristics – A Survey. *Simulation and Gaming*, **40**, 726–751.

Müller, J-P. *Mimosa*. http://mimosa.sourceforge.net.

Vangheluwe, H.L.M., & Vansteenkiste, G.C. *The Cellular Automata Formalism and its Relationship to DEVS*.

Wickler, G., Tate, A., & Potter, S. 2007. Integrating Discrete Event and Process Level Simulation for Flexible Training in the I-X framework. *In:* Van der Walle, B., Burghardt, P., & Nieuwenhuis, C. (eds), *Proc. ISCRAM 2007*.

Zeigler, B.P., Praehofer, H., & Kim, T.G. 2000. *Theory of Modeling and Simulation 2nd edition*. By Bernard P Zeigler, Herbert Praehofer, and Tag Gon Kim.

## Authors Biography

**Terry Bossomaier** is the Director of CRiCS, the Centre for Research in Complex Systems. His research interests range from neural networks to agent-based modelling. He is currently working on various aspects of serious games and artificial intelligence for game play.

**James Tulip** recently completed his PhD in image processing, but also works on computer game engine software and the design of computer games.

**John Carroll** is the Deputy Director of CRiCS and works in many aspects of new media and computer games. He is particularly interested in the use of applied drama in serious games.

**David Cameron** is a former journalist now working in serious games and public relations research. He coauthored with John Carroll the 2006 book, *Real Players*.