

MNIT Deliverable: Year 1

WP 1 - Experiments

ABSTRACT

This report summarises year one experimental activity, comprising: experimental approach and methodology, analysis and networking tools, and network and user analysis of the principle year 1 experiment, Out Of This World. Appendices are: MASSIVE-2 network packet format, report of porting activities, and documentation of analysis tools.

Document ID	MNIT-year1-1
Work Package	WP1: Experiments
Status	Final
Version	1.0
Date	27/1/1999
Author(s)	Chris Greenhalgh

Task	A1.1
-------------	------

Table of Contents

1	INTRODUCTION	4
2	EXPERIMENTAL METHODOLOGY	5
2.1	Trial components	5
2.2	Available resources	6
2.3	Experimental issues	7
2.3.1	Elements of a trial	7
2.3.2	Possible trial outcomes	7
2.3.3	What kind of analysis can be performed?	8
2.3.4	What issues can be addressed?	8
2.3.5	Concrete proposals	9
2.3.6	Issues address in OOTW analysis	9
3	EXPERIMENTAL TOOLS	11
3.1	Network simulation	11
3.2	Network traffic capture	11
3.3	Network traffic analysis	12
3.4	Application and user analysis	12
4	OOTW NETWORK AND USER ANALYSIS	13
4.1	Show 2 network traffic	13
4.1.1	Multicast traffic components	14
4.1.1.1	Multicast traffic and show phase	16
4.1.2	Unicast traffic components	17
4.1.3	TCP traffic components	19
4.2	Network traffic and levels of participation	20
4.3	User activity: movement rates	22
4.3.1	Overall movement rates	22
4.3.2	Movement and levels of participation	23
4.3.3	Movement and the management interface	24
4.3.4	Movement and teams	25
4.3.5	Movement and shows	26
4.3.6	Correlation of movement	26
4.3.7	Movement and phases	27
4.4	User activity: audio participation rates	29
4.4.1	Silence detection	29
4.4.2	Overall audio rates	30
4.4.3	Audio and levels of participation	30
4.4.4	Audio and teams	31
4.4.5	Audio and shows	32
4.4.6	Correlation of audio	32
4.4.7	Audio and phases	34

4.5	Summary	34
4.6	Conclusions from OOTW	36
5	CONCLUSIONS	37
6	APPENDIX A: PORTING AND INSTALLATION OF MASSIVE-2	38
7	APPENDIX B: ANALYSIS TOOLS DOCUMENTATION	39
7.1	Network analysis tools – packets	39
7.1.1	Tcpdump2fs	39
7.1.2	Tcpdump2data	40
7.1.3	Tcpdump_compare	40
7.1.4	Find_rtt	40
7.2	Flow analysis tools	40
7.2.1	File formats – fs, flow	40
7.2.2	File formats – rules	41
7.2.3	Fs2flows	42
7.2.4	Flowfilter	43
7.2.5	flow2data	44
7.2.6	Summary	44
7.3	User analysis tools	45
7.3.1	File format	45
7.3.2	Generating files	45
7.3.3	Sort_by_time	45
7.3.4	Process_simple	46
7.3.5	Process_correlation	46
8	REFERENCES	48

1 Introduction

The experimental workpackage is WP1. Year 1 is nominally concerned with installing MASSIVE-2 on the BT Future's Testbed, establishing the initial infrastructure and performing initial experiments.

Following discussions with BT in May 1998 it was decided to suspend plans for work on the Future's Testbed until further notice. Experimental activity in year one has focused on LAN and simulated wide-area infrastructures. Experiments on the Future's Testbed will be resumed as and when appropriate to current experiments and development issues. Porting and installation work on MASSIVE-2 is summarised in Appendix A.

The main body of this document is structured as follows:

- Section 2 describes the experimental philosophy and methodology which we have applied.
- Section 3 describes tools which we have used to simulate varying network characteristics and to log and analyse network traffic from experiments.
- Section 4 presents the analysis of network traffic and corresponding user activity from Out Of This World, the primary year 1 experiment (jointly with the EC-funded eRENA project).
- Section 5 has final conclusions.
- Appendix A describes porting and installation work carried out for MASSIVE-2 in the context of earlier experimental plans.
- Appendix B includes documentation for the network and user analysis tools referred to in section 3.

2 Experimental methodology

This section describes the framework and approach which we have used in designing experiments and trials, and in analysing them and structuring their results. This framework is developed from our initial experiences in the BT/JISC-funded Inhabited The Web (ITW) project.

2.1 Trial components

Trials and experiments with collaborative virtual environments (which are the interactive basis for Inhabited TV) involve many complex and interacting elements. To make this problem more tractable we divide up a trial into a number of layered components, each of which can (to a certain extent) be considered in isolation. The interactions between these components are largely localised at layer boundaries, although secondary influences extend from every component to every other component.

Using this framework it is possible to construct models for different layers (e.g. from systematic measurements, or simulations), plug these models or measurements together, and so build a complete predictive and analytical model for a CVE. For example:

- We can observe the patterns of activities of users in a CVE to gather information about general patterns of activity.
- We can observe and analyse the network traffic generated by a CVE system in controlled situations, in order to build a general network behaviour model for the system. Such a model predicts (in principle) the network traffic which results from any specific virtual activity (e.g. one step of movement, one second of speaking).
- We can combine both of these to predict, for any number of users, the network traffic which we would expect to see – the user model describes what we expect the users to do, and the system model describes the corresponding communication requirements.

Using this combined model we can answer (at least in principle) questions such as:

- For a given numbers of users in a given system, how much bandwidth would be required.
- For a given available bandwidth (or CPU capacity) in a given system, how many users could be supported.

With extensive analysis and multiple trials, we can begin to characterise the influences of other components, which typically have more subtle effects or are harder to characterise or model in a generalised form.

Figure 1 shows the seven components into which we divide a trial or networked experiment with a virtual environment system. These components are explained below.

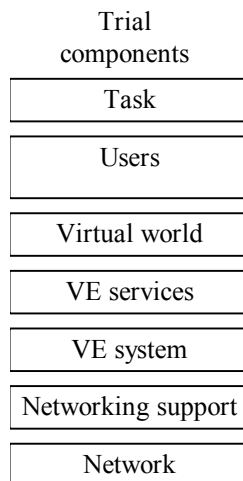


Figure 1 components of a trial

- The network is the actual communication infrastructure, such as Ethernet, ATM, ISDN, etc.. We normally restrict our consideration of this component to individual (unreliable) data packets, typically IP packets in the systems we have used to date. We are interested in aspects of the network such as bandwidth, delay and loss rate.
- The networking support includes network transport protocols and similar middleware functions, such as reliable multicast. We are interested in sessions, semantics (e.g. unicast vs. multicast) bandwidth, delay, reliability and other aspects of quality of service.
- The Virtual Environment (VE) system makes use of the (nominally) generic networking support facilities to create a distributed virtual environment. Year 1 has concentrated on MASSIVE-2 as the sole VE system (ITW used MASSIVE-1). The VE system supports the creation of worlds, sharing of data and updates and other communication and interaction.
- The VE services are the facilities and affordances which the VE system provides to users and applications. For example, users of MASSIVE-2 are embodied in a shared 3D virtual world and can navigate in 6 degrees of freedom, talk to one another (using packetised network audio), move artefacts about within the virtual world and “sleep” (a simple graphical gesture in which their embodiment lies down).
- The virtual world is typically specific to a given application or trial and includes both static content (e.g. scenery, walls) and active artefacts which may comprise the interface of an interactive application (e.g. the game objects – such as the “space frogs” in Out Of This World). These provide the particular audio-graphical context in which the trial virtually occurs.
- Our focus is on systems for human use, and especially collaborative use by numbers of simultaneous participants. Different users will have different expectations, skills, aptitudes and interests, which will have an impact on any trials or experiments in which they participate.
- Multiuser Virtual Environments can be used for a wide range of purposes (e.g. tele-conferencing, co-design, entertainment, education), and may include many different sub-tasks (e.g. giving a talk, reaching a consensus, interacting with a visualisation). The task (taken in a very broad sense) will affect what users do and how the system is used.

2.2 Available resources

In order to characterise or model these layers it is useful and sometimes essential to obtain information from real use (i.e. from trials and experiments). Figure 2 shows the same trial components, together with the corresponding information resources which can be captured during a trial, to inform analysis and modelling. These are described below.

Trial components	Persistent data (for analysis)
Task	Direct observation, video, questionnaires, interviews, task output
Users	
Virtual world	Application logs
VE services	Service logs
VE system	System logs
Networking support	
Network	Network logs

Figure 2 components of a trial and consequent resources for analysis

- We have made extensive use of network logging, using the tcpdump packet logging tool as described in section 3. We have captured network traffic logs for OOTW, as analysed in section 4.

- A VE system may be instrumented (typically by hand, given source code access) to generate system and service log files. At the system level we have instrumented MASSIVE-2 to log information about rendering and some other internal details.
- At the services level we have instrumented MASSIVE-2 to log information about user movement, sleeping, mouth animation (all services) The world manager process in OOTW was also instrumented to log VE service information for the virtual world, as a centralised adjunct to logging from individual user processes. We have existing and related analyses of use of VE services (i.e. user activity) from ITW trails with MASSIVE-1, and EC-funded COVEN project trials with dVS and DIVE. The analysis of OOTW is reported in section 4.
- Specific applications can be instrumented to record application-specific activities, for example details of user interactions and requests. Typically, this will be done by the application programmer. For example, we instrumented much of the OOTW management interface in order to facilitate subsequent analysis of the event (e.g. recording the triggering or phases or the use of direct control).
- Analysis of users and tasks typically relies on methods derived from psychology and the social sciences, such as observation, video recording, questionnaires, interviews (structured or open) and measures of task-related output (e.g. how accurately or quickly a set task was performed). In the context of MNIT we have mainly used these resources in a supporting role, for example when considering anomalies in other data sets; they are used more directly in other projects (e.g. eRENA).

2.3 Experimental issues

This section gathers together a number of issues related to the conduct of trials and experiments: what is required to conduct a successful trial; what information and outcomes a trial can generate; what kind of analyses can be performed; what issues can be addressed; and a number of possible experimental objectives in the context of MNIT. This is intended to provide a resource to inform future consideration of trials, and for determining whether and what trials may be appropriate within the context of this project.

2.3.1 Elements of a trial

A successful network trial draws on the following elements (where “appropriate” is interpreted with reference to the desired research hypotheses):

- An appropriate number of appropriately skilled and interested/involved users with simultaneous access to appropriate hardware, software and networking.
- A task or activity which will cause the users to act appropriately.
- Software support for application and system logging.
- Facilities for network traffic logging.
- Video capture of one or more users.
- A virtual environment appropriate to the task.
- A stable and appropriate VE system.
- Stable and reliable network communication of sufficient bandwidth and small enough end-to-end delay for the duration of the trial.
- Collection of logs and archival at the end of trial.
- Possible post-trial questionnaires and other feedback.

2.3.2 Possible trial outcomes

Any given trial has a range of possible outcomes, some of which may be more desirable than others:

- Something(s) may fail:
- the network may be “inadequate” (bandwidth, loss, delay, multicast support);
- the system may crash or be unstartable at one or more sites;

- the system may not provide appropriate services or behaviour during the trial;
- the virtual environment may prove inappropriate;
- the users may get lost or otherwise fail to establish or sufficiently coordinate the task activity;
- logging (video recording, etc.) may fail or not be performed;
- logs (videos, questionnaires, etc.) may be lost (due to system or human error or due to failure to collect them);
- logs may turn out to have wrong, inadequate or inappropriate information in them.
- Alternatively, something(s) may NOT fail (see above), implying partial validation of that aspect of the trial.
- It may all succeed and appropriate data be made available for subsequent analysis.

2.3.3 What kind of analysis can be performed?

A range of possible analyses can be performed, including:

- Ethnographic (sociological and psychological) studies of trials and video transcripts.
- Automated analysis of network traffic.
- Automated analysis of application, service and system event logs.
- Statistical or qualitative (psychological) analysis of questionnaires, interviews and task output.
- Qualitative self-assessment (less rigorous but useful in the early stages).

In MNIT we have made most use of the second and third (e.g. in the analysis of OOTW). In other projects and contexts we have employed all of the above (often in collaboration with exponents of those approaches).

2.3.4 What issues can be addressed?

From the above, a trial might be considered as the tuple (Task/Occasion, Users, VirtualWorld, VEServices, VESystem, Networking, Network).

- We might attempt to establish the adequacy of sufficiency of one or more components in the context of the trial:
- does it (task, system, network, etc.) work (for how long, under what circumstances) as measured by (task, logs, network, user feedback, video, etc.)?
- is it (all?) enjoyable (for the user, application programmer, system administrator, etc.)?
- We might attempt to establish the characteristics of one or more components of the trial in this particular context. E.g. by manual or automated analysis of logs as in the ITW trials. E.g.
- what are the characteristics of user behaviour and activity;
- what degree and rate of network and system reconfiguration requests occur;
- what is the resulting traffic like (bandwidth, burstiness, etc.).
- We might look for breakdowns, failures, mismatches, limitations in one or more components or the relationships between them (implying scope for new designs, alternative approaches, etc.).
- do users make errors or take additional (unnecessary) steps to complete operations;
- does the system (world, service, etc.) support normal (expected, efficient, etc.) working practice;
- do users understand what is happening (and do they - rightly or wrongly - believe that they understand).
- We might explore the effect of varying one or more components of the trial, e.g. dependence on task, users, system, etc. Influence might be assessed in terms of any subset of the trial components.
- is a CVE better for task A or B (and how much);
- is it better with users of type A or type B;
- is a system using technique A better than one using technique B;
- is a network with characteristics A better than one with characteristics B;

- is a network (or system or service or world) with characteristics A “adequate” (as measured by... task performance, say);
- does variation in component X (task, users, etc.) have a large or small effect on component Y (user behaviour, network traffic, etc.), i.e. how carefully must it be considered.
- We might attempt to compare the use of a CVE with a completely different system, method or approach:
- is it better (task performance, user feedback, bandwidth, etc.) than method X for task A, users B, etc.

2.3.5 Concrete proposals

Focusing on automated analysis of user activity and networking we can suggest the following possible trials as being of potential relevance to this project, expressed in terms of controlled variables, outputs and (partial) hypothesis.

Controlled variables:

- varying numbers of simultaneous users;
- different QoS allocation schemes;
- different consistency characteristics;
- (artificial) network latency, jitter, loss rate (also wide-area vs local area; Mbone vs application bridged);
- different video (and other) applications/services;
- different QoS control and feedback techniques.

Outputs:

- network bandwidth, burstiness, etc.;
- system dynamics;
- user activity/behaviour;
- user perception/satisfaction;
- measures of divergence/inconsistency;
- measures of interaction latency/responsiveness/failure.

Pre-hypotheses:

- requirements/network characteristics of system/technique X with N users is $F(X,N)$;
- perceptual quality (usability, accuracy, etc.) of allocation scheme Y is $F(Y)$;
- consistency type Z has importance $F(Z)$ (as a function of latency, loss, etc.);
- application/service A is useful/enjoyable;
- application/service B has characteristics $F(B)$;
- requirements for latency, jitter, loss for task/application C are $F(C)$;
- QoS control/feedback scheme D has effect $F(D)$.

2.3.6 Issues address in OOTW analysis

The OOTW analysis address the following issues with regard to network bandwidth:

- Effects of levels of participation.
- Effect of choice of medium.
- Effect of phases of the event.
- Effects of using silence suppression.

With regard to user activity – audio and movement – it addressed the following issues:

- Effects of levels of participation.
- Effects of different shows and different teams.

Multimedia Networking for Inhabited Television - University of Nottingham - 1999

- Correlation between users' activities.
- Individual variability.

3 Experimental tools

This section describes the tools which we have employed (and created) in support of experimentation and analysis. These comprise a general-purpose experimental infrastructure, partly developed in MNIT and partly in other previous and current projects (e.g. ITW, COVEN). These are dealt with under the headings of: network simulation; network traffic capture; network traffic analysis; application and user analysis.

3.1 Network simulation

We have made use of the “dummynet” kernel-based network emulation tool, created by Luigi Rizzo at the Dipartimento di Ingegneria dell’Informazione, University of Pisa [1][2]. This comprises a set of kernel patches for FreeBSD which simulate the effects of network delays and bandwidth limitations by internally delaying and discarding IP packets as they transit through the kernel.

This software is now in its second version, which we have recently installed and configured within the CRG local network for future experiments. This second version is run on a PC acting as a router, and uses the IP firewall facilities of FreeBSD to selectively delay or bandwidth limit flows of IP packets (selected according to some subset of protocol, source and destination addresses). This allows the router to simulate multiple connections of varying characteristics, including variable delay, bandwidth and packet loss rate.

We extended and employed the first version of dummynet for the consistency experiments with MASSIVE-2 performed by Ivan Vaghi (reported under work package 2, architecture with consistency). It has also been used for HIVE project experiments. The first version allowed only a single restriction to be placed on all IP traffic in transit through the machine’s kernel. We used this with IP forwarding to create a packet “reflector”, to which IP packets could be sent, delayed, and then forwarded to their final destination. This was done by adding appropriate static IP routes in the experimental machines, so that IP packets destined to other experimental machines would be sent via the network simulator machine. This is shown diagrammatically in figure 3.

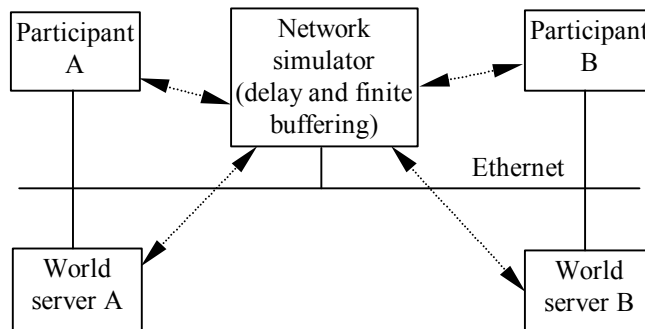


Figure 3 initial use of dummynet version 1 for network delay experiments

3.2 Network traffic capture

We have used the publicly available tcpdump tool [3] to capture network packets. This normally relies on the use of bus-based networks, typically Ethernet and FDDI, to capture packets not otherwise transiting the logging machine’s kernel. All experiments to date have been conducted with one or more LANs, and consequently open to this form of logging. Care must be used with twisted-pair cabling and Ethernet switches: packets between other machines are typically not seen by the logging machine. Ethernet hubs must be used, or the switch must be configured to send all packets to the logging machine, or logging must be performed on each machine individually.

Tcpdump captures a configurable number of bytes from the front of each network packet, including IP and UDP/TCP packet headers. Packet traces are initially captured to log files, for subsequent off-line

analysis. Tcpcap also (optionally) provides human-readable information about each packet, which can be helpful in initial debugging and testing of capture.

3.3 Network traffic analysis

We have recently (for the OOTW analysis and a related COVEN project analysis) created a suite of network traffic analysis tools. These are based on the notion of flows: sequences of packets matching a common pattern, which are assumed to be related. These tools are currently specific to the IP protocols, and most effective with UDP and TCP traffic (the majority of IP traffic). We have made use of, and extended, the network flow analysis tool FS2flows [4][5] from the US National Laboratory for Applied Network Research (NLNR). This tool processes a log of IP packets and summarises it in terms of traffic flows (sequences of related packets) and their timing and bandwidth. We have extended this tool to generate time-series records for each flow; by default it only creates information on each completed flow.

The finest grain flow which we work with is defined by the 5-tuple:

(IP-protocol, IP-source-host, IP-dest-host, IP-source-port, IP-dest-port)

Note that the ports are only valid for UDP and TCP protocols. A single flow is a sequence of packets from one machine and port to another machine and port with a given IP protocol number. For example, this corresponds to one half of a TCP connection, or a flow of multicast packets from a single source.

In addition to the identifying basic flows, we have created a flow-summarisation program (flowfilter) which draws on ideas from NeTraMet [6] and the current IETF work on traffic measurement in the Realtime Traffic Flow Measurement working group [7]. This allows the analyst to progressively aggregate information from different flows, according to patterns based on the above tuple. For example, all traffic to or from a given machine could be combined, or all web traffic (i.e. flows to and from default port 80) could be combined. There are also simple scripts to summarise (tabulate) information about the composite flows, and to generate data suitable for plotting with tools such as Excel or gnuplot.

Further documentation for these tools can be found in appendix B. The results of their application can be seen in section 4's analysis of OOTW.

3.4 Application and user analysis

In the prior ITW project we created a number of tools to analyse application log files (in that case from MASSIVE-1) and to extract statistics on the use of the VE services, such as movement, and speaking. These have been updated and extended to some extent in support of the analysis of OOTW. For this and other projects we have also created a range of additional programs and scripts which convert logs from different systems (e.g. MASSIVE-1, MASSIVE-2, OOTW manager, dVS, DIVE) into a common form for analysis.

The main tasks to which we have applied these tools are:

- Determining the fraction of time for which users are moving (or speaking or manipulating objects).
- Determining the extent to which they are moving (or speaking or manipulating objects) at the same time as other users.
- Visualising these activities, and transitions between different worlds (currently applicable to MASSIVE-1 and DIVE).

These give a means of constructing simple statistical models of low-level user activity, or equivalently the demands likely to be placed upon the VE services.

These tools are documented in appendix B, and their use can be seen in the analysis of OOTW, in section 4.

4 OOTW network and user analysis

This section presents our analysis of Out Of This World, the principal year one experiment. The event itself is described in an accompanying deliverable.

This analysis is based on data from the four performances of out of this world, staged on the 5th and 6th September 1998. In particular, we make use of the following:

- logs of network traffic data captured using the tcpdump tool; and
- log files from the world manager process, which records the initiation of phases in the game and also records the presence and movement in the virtual world of all users.

To determine the exact timings of the performances we made use of video recordings captured from a monitoring machine, plus clock synchronization information from the PLOD distribution platform.

We begin by examining the network traffic from a particular show (show 2) in detail. In particular, we:

- explore the network traffic from that show, identifying the key types of traffic and the form of the performance's bandwidth requirements; and
- consider the impact of show phases on the network traffic; and
- compare the bandwidth requirements of different traffic sources according to their level of participation (host, captain, inhabitant, camera);

We then move from direct consideration of network traffic to consider user activity (in particular movement and speaking) during the shows. In particular, we:

- analyze the overall rates of moving and speaking as seen by the network, and relate these to previous studies;
- characterize the impact on speaking and movement of the different levels of participation employed in Out Of This World;
- compare the levels of activity of the two teams (one with male member and other with female members);
- compare the levels of activity in different shows; and
- explore the apparent effects of phases and activity constraints on user activity.

Finally, there is a summary of our findings, and some initial conclusions and proposals specific to MNIT.

4.1 Show 2 network traffic

In this section we explore the network traffic from show 2. Figure 4 shows the total bandwidth for this show, calculated at one minute intervals. It also shows the top-level traffic breakdown into multicast UDP traffic, unicast UDP traffic and TCP traffic. The traffic trace begins at 20:03:08 on the 5th September 1998, just before the world and all users are restarted for the show. The actual broadcast from the world runs from 20:43:02 (time 2394) to 21:26:52 (time 5024) - traffic is shown for a further 5 minutes after the end of the show.

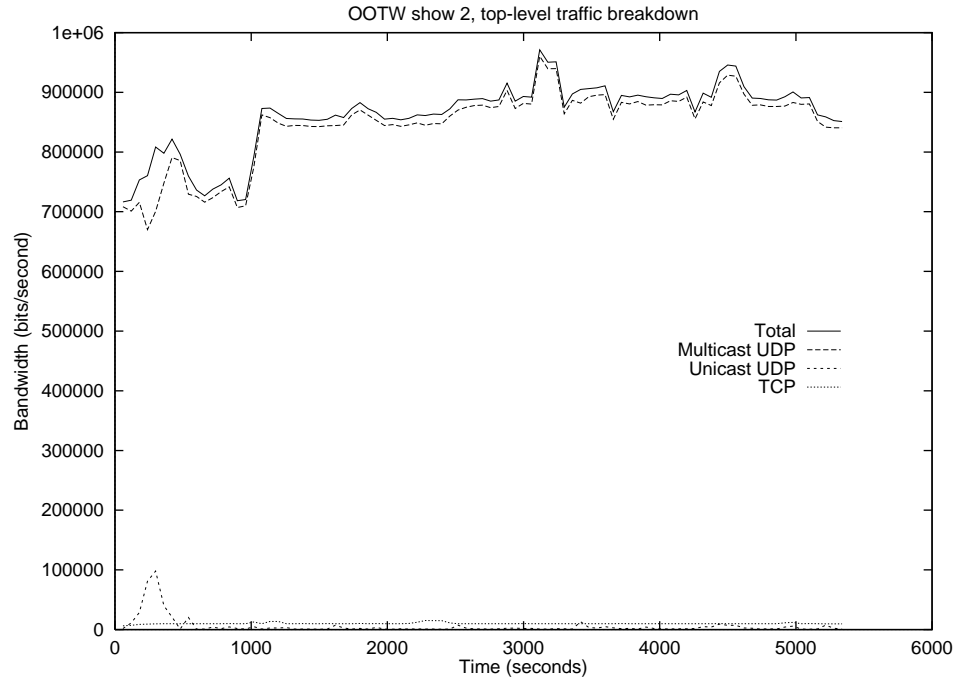


Figure 4 top-level traffic breakdown for show 2. (show2.toplevel.eps)

The table below summarizes the traffic breakdown.

Type	Bytes	%	rate bit/s
UDP-MC	564093044	98.2	845083.2
TCP	6759358	1.2	10126.4
UDP-UC	3737055	0.7	5598.6
Other	9380	0.0	14.7
Total	574598837	100.0	860822.2

We observe from the figure and the table that:

- The network traffic is consistent throughout, at around 860 Kbits/second.
- The vast majority of this traffic (98.2%) is multicast UDP traffic. This in turn comprises audio, video and application components, which are explored below.
- The unicast UDP traffic is concentrated at the start of the session, between time 0 and time 600; this corresponds to starting up the world and the users, when unicast is used for state transfers. There are smaller bursts of unicast UDP traffic throughout the session; unicast is also used for ensuring reliability of multicasting and for various direct interactions (e.g. applying management constraints to users).
- There is an almost constant background level of TCP traffic at around 10 Kbits/second. This is considered below.

The following subsections consider the multicast, unicast and TCP traffic components in more detail.

4.1.1 Multicast traffic components

Using destination port numbers we can identify the main traffic types within the total multicast traffic. Their contributions are summarized in the table below and plotting in figure 5.

Type	bytes	%	rate bit/s
Audio	523160128	91.0	783760.5
Video	22709952	4.0	34804.5
CVE	18219904	3.2	27295.7
Other MC	3060	0.0	4.7

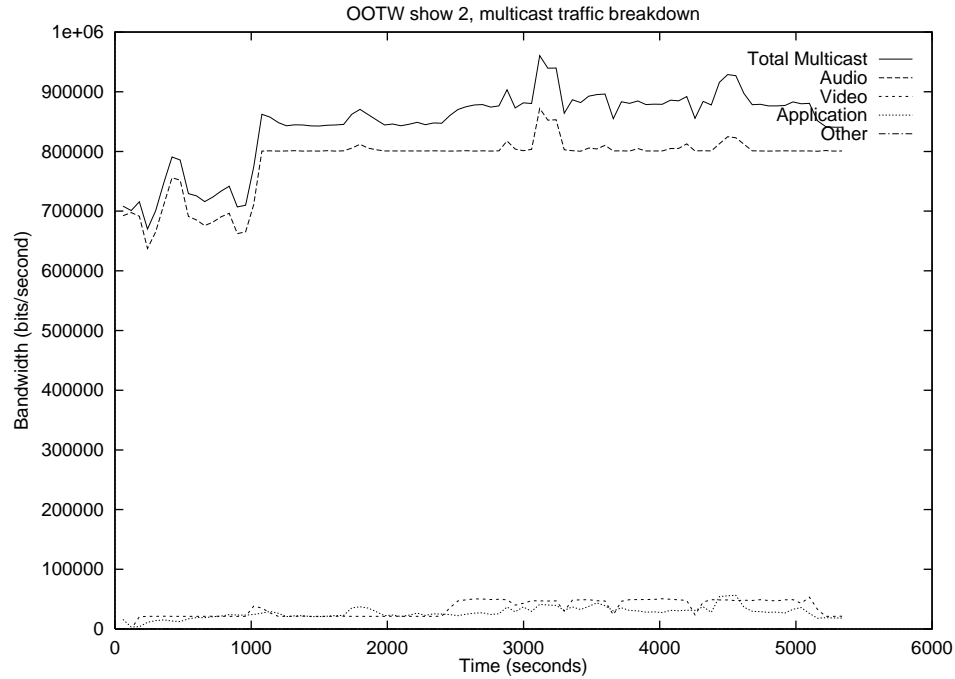


Figure 5 multicast traffic components for show 2. (show2.mc.eps)

Considering each component in turn:

- After time 1000 (before the start of the show) the audio traffic level is very stable at about 800 Kbits/second throughout the show. There are small bursts of additional traffic, for example around time 3200 and 4500. The background audio level corresponds to real-time audio from the 11 users (host, two captains, eight inhabitants). This level is so consistent because the normal silence detection mechanism was disabled. This was done on site to avoid audible artefacts from the operation of the mechanism. Consequently, every user sends audio traffic continuously. The localized bursts of traffic are sound effects generated during the games.
- The video traffic moves between two steady levels of about 21 Kbits/second and 49 Kbits/second. There is a single video stream for the host. This uses JPEG compression for each frame, and consequently the resulting traffic is greater for images which contain more complexity. In this case, the lower bandwidth corresponds to times when the host is not in front of the camera (e.g. before the event starts around time 2400), while the higher bandwidth is when the host is present, i.e. for most of the game. During the end-of-game score review the host screen is not visible to the inhabitants and the host appears to move away from the camera, accounting for the dips around time 2900, 3300, 3700 and 4200.
- The application (CVE) traffic accounts for changes in the virtual world, such as: users moving, the appearance and disappearance of speech balloons, and the activity of objects within the worlds (e.g. the space frogs). This has a fairly consistent background level of about 20 Kbits/second, rising to a maximum of 56 Kbits/second (one minute average) around time 4500 (the race). We consider the relationship of this variation to the phase structure in more detail below.

4.1.1.1 Multicast traffic and show phase

Figure 6 shows the four components of the multicast traffic which are most affected by the progression of the show itself for the main period of the show. For reference, the main stages of the show are as follows:

- Pre-show football match, time 1700-1900;
- Start of show, time 2400;
- Football match during introduction, time 2800-2890;
- Move to arena 1, time 2945-3020;
- Space frog game, time 3050-3240;
- Move to arena 2, time 3290-3370;
- Falling fish game, time 3400-3600;
- Move to arena 3, time 3670-3760;
- Quiz game, time 3770-4200;
- Move to arena 4, time 4260-4350;
- Race, time 4370-4570;
- Wobblespace interaction, time 4895-4955;
- End of show, time 5025.

There are many more actual management phases than this, since each of these stages typically comprises several phases (e.g. start of stage, active phase, move to end of stage, move to next stage).

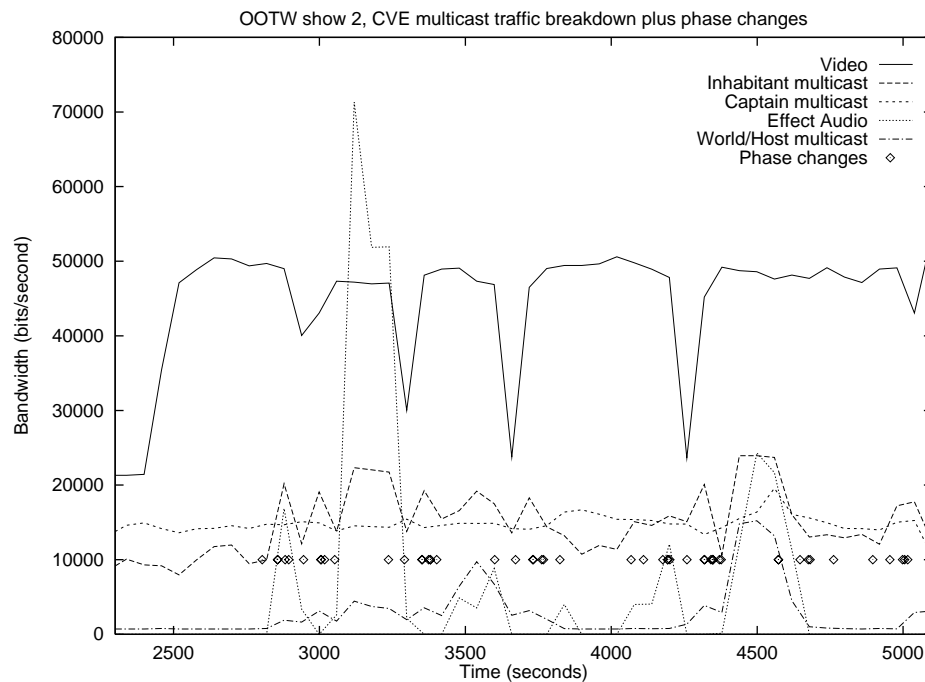


Figure 6 breakdown of CVE multicast traffic plus sound effect audio, with phase changes shown for comparison. (show2.mc-world.eps)

For the traffic components shown we note the following.

- The video of host, as noted above, increases when the host is in front of the camera - at the start of the show, and decreases again during each end-of-game debriefing (times 2945, 3290, 3670 and 4260).

- The sound effect audio occurs during the football game (the ball "booings" when it is hit), the space frog game (they "croak", and "squeal" when impaled), the fish game (fish "chime" when harvested), the quiz (the rings give a fanfare when passed) and the race (the cones "crash" when hit). The correspondences to phases is thus very clear.
- The team captains produce a very consistent level of traffic at around 15 Kbits/second, total. This is dominated by the continuous stream of position updates which are generated from the use of magnetic trackers to control their embodiments (see the user movement analysis in section 3 for more details).

The remaining CVE application traffic are also shown in figure 7 at a finer level of temporal detail (10 second intervals).

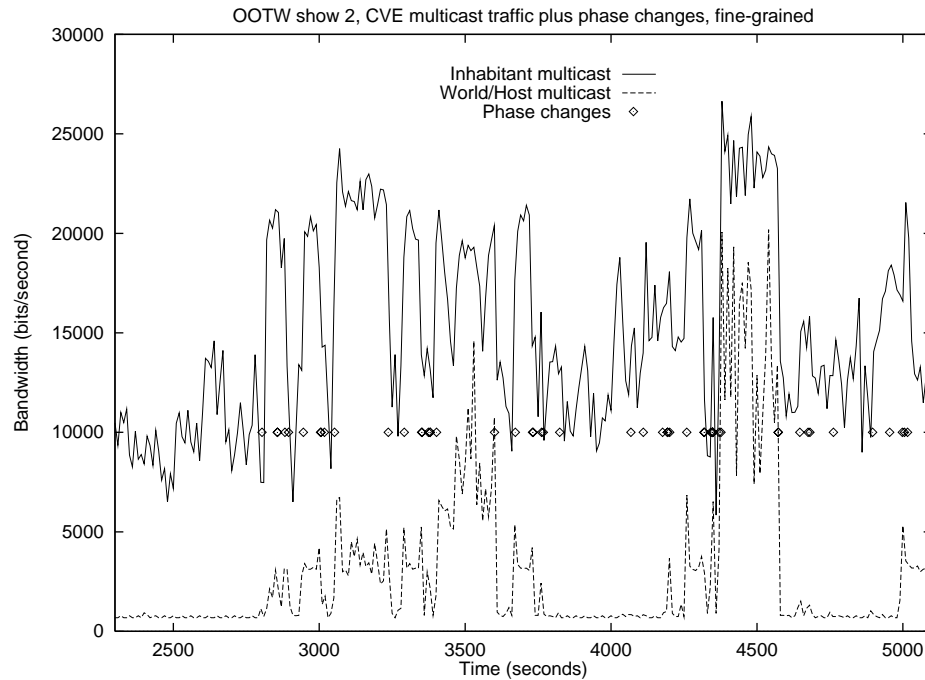


Figure 7 CVE multicast traffic with phase changes shown for comparison. (show2.mc-world-10.eps)

We note that:

- The world/host multicast traffic has a constant background level around 7000 bits/second; this is keep-alive messages for the world to show that it has not crashed. There are peaks in activity during the games corresponding to the activity of world objects such as the football, the space frogs, the platforms (in the fish game) and the cars in the final race. Again the relationship to phases is direct and simple. There are also blocks of traffic during movements between arenas corresponding to the movement of the host's embodiment.
- The inhabitant application multicast traffic exhibits quite a fine-grained relationship to the phases of the game, although the overall variation is still relatively limited, e.g. less than 10 Kbits/second at times in the relatively controlled entries to and exits from arenas (e.g. times 2900 and 3050), rising to about 25 Kbits/second during the race (time 4370-4570). However most phases exhibit substantial variability, for example during the quiz (time 3770-4200) the inhabitant application bandwidth ranges from 9 to 19.5 Kbits/second. In the later section which examines user activity we will look again at the relationship between phases and user activity (and its network implications).

4.1.2 Unicast traffic components

We have already noted that the unicast UDP traffic is concentrated at the start of the period, before time 600, where we would expect state transfers to be taking place. The unicast UDP traffic in

MASSIVE-2 has three roles: state transfers, ensuring multicast reliability, and point-to-point communication (e.g. the world manager imposing constraints on users and objects).

In order to estimate the amount of reliability-related traffic we consider separately:

- unicast traffic to and from the various node manager processes, which handle world and object name resolution and to monitor clock offsets between machines on the network (for post-event compensation and analysis);
- the unicast traffic which flows to and from the world host, which should include world state transfers plus all management traffic (and associated reliability-related traffic); and
- unicast traffic between other machines, which will include initial state transfers (for user embodiments), but only reliability-related traffic after this.

The breakdown is shown in figure 8 and in the table below. The application multicast traffic is also shown for comparison, as are the moments at which phase changes are triggered (shown as points on the graph - vertical position has no inherent meaning for these).

Type	bytes	%	rate bit/s
MC CVE	18219904	3.2	27295.7
World	2340236	0.4	3506.0
Rest	925035	0.2	1385.8
NodeMgr	471784	0.1	706.8
Total	3737055	0.7	5598.6

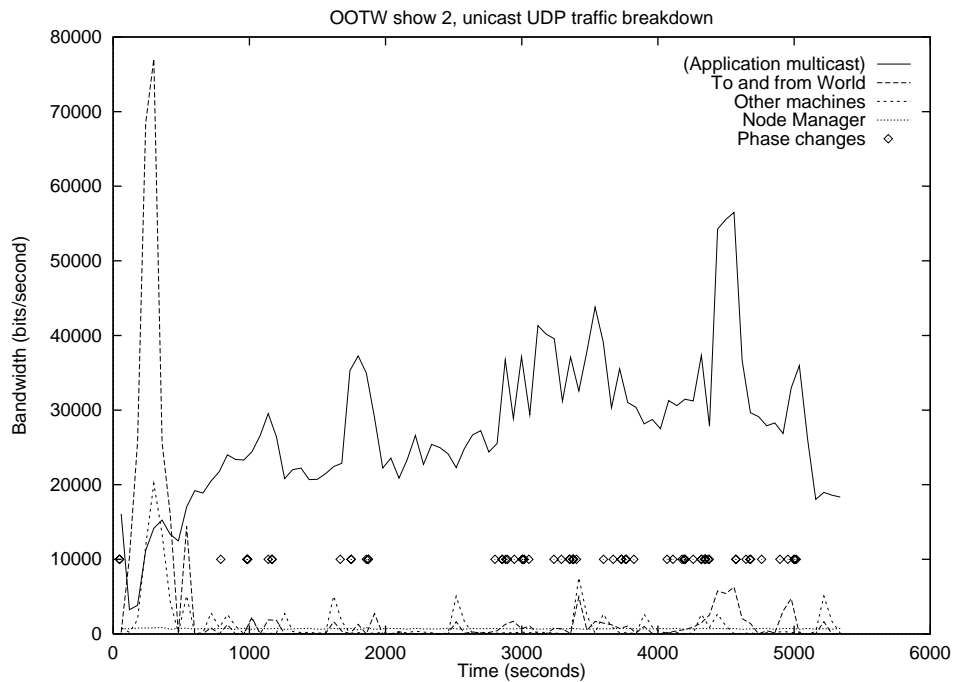


Figure 8 traffic breakdown for unicast UDP traffic, show 2 (including application multicast traffic and show phase changes for comparison). (show2.uc.eps)

We make the following observations:

- State transfers appear to be (and we would expect to be) concentrated in the first 600 seconds. This period accounts for 60% of the total UDP unicast traffic.
- There are some clear examples of phase-related unicast traffic, between the world machine and other machines, for example around times 1200, 1900, 2900.

- There is also clearly some residual reliability-related traffic. This appears to be of the order of 10% of the unicast traffic, or equivalently less than 1% of the corresponding multicast traffic. It is quite peaky in character, though it bears no clear relation to variations in the multicast traffic.
- Node manager traffic, primarily timing checks, accounts for around 12% of the total unicast traffic. It runs consistently throughout at around 700 bits/second.

4.1.3 TCP traffic components

We have already seen from the top-level breakdown that the TCP traffic is a very consistent background level of around 10 Kbits/second. If we examine this in more detail we can identify the following components:

- Remote login traffic, used by the system administrator to monitor the progress of the event. The constant rate indicates that they are receiving a relatively constant rate flow of status information. There is one additional burst of remote activity around time 1000.
- Process watchdog, which was a TCP-based program to monitor the liveness of the various programs comprising the complete distributed system. This was intended to assist in detecting and diagnosing process failures, though in fact the software was stable throughout. This is essential a continuous flow of status messages from all active processes.
- A block of TCP traffic around time 2300 which is the network communication for interfacing wobblespace to the pong game in the opening sequence.
- Wobblespace events, around time 4900, which is the network communication interfacing wobblespace to the virtual world to drive the inhabitants up the tubes to spaceship.
- A little video membership monitor traffic around time 100-400 as new user processes start up and register with the video service.

This traffic breakdown is shown in figure 9 and summarized in the table below. This accounts for all of the observed network traffic.

<i>Type</i>	<i>Bytes</i>	<i>%</i>	<i>rate bit/s</i>
Login	4400621	0.8	6592.7
Watchdog	2148560	0.4	3218.8
Other TCP	154105	0.0	266.8
Wobblespace	29836	0.0	47.9
MembMon	7548	0.0	11.8

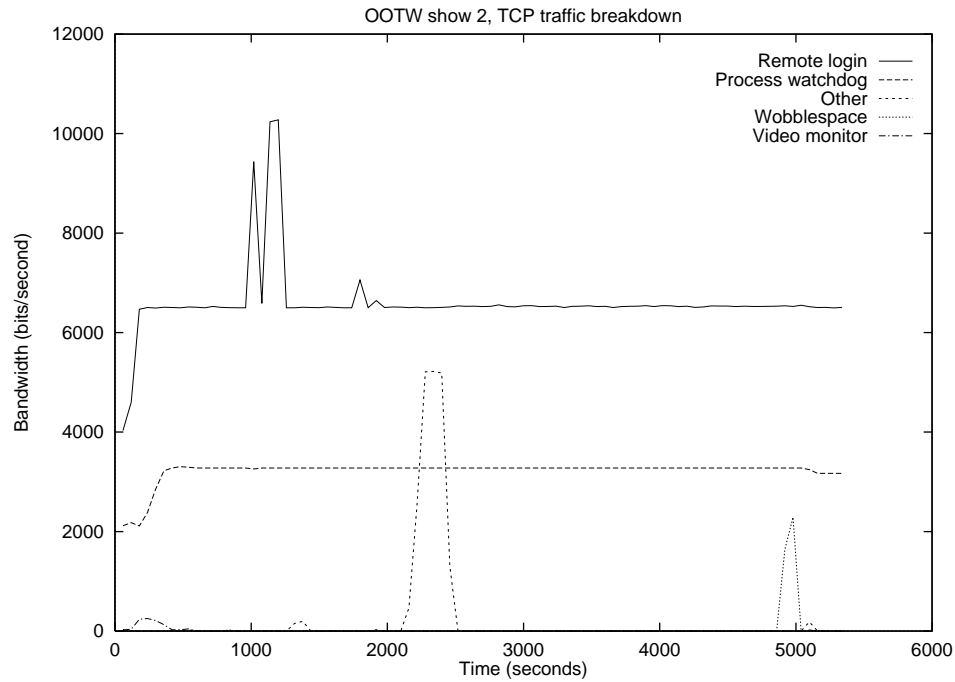


Figure 9 breakdown of TCP traffic, show 2. (show2.tcp.eps)

4.2 Network traffic and levels of participation

In the previous section we explored the network traffic from show 2, and showed how it breaks down into multicast UDP, unicast UDP and TCP components. We further showed how these break down into their constituent activities. In the case of the application component of the multicast traffic we have shown how this varies (to a limited extent) with the various phases of the show as it progresses.

In this section we relate network traffic to the different levels of participation used in OOTW; these were:

- Host, the show compere, embodied using live audio plus video (the only participant with video);
- Team captain, using an immersive tracking-based interface with live audio;
- Inhabitant, using a desktop joystick-controlled interface with live audio; and
- Camera, using a desktop, mouse-based, interface, not normally visible in the virtual world and having no live audio.

Firstly, there is simple relationship between level of participation and media, which in turn has a direct impact on the network. Specifically:

- Only the host has a video stream, which consumes around 49 Kbits/second when they are on-screen (around 20 Kbits/second when they are not). So the host's bandwidth requirements are of course 49 Kbits/second greater than an equivalent non-video participant.
- The host, captains and inhabitants all have real-time audio which contributes 72.8 Kbits/second of audio network data. Recall, that in OOTW silence detection was disabled to improve the perceived quality and reliability of the audio channel, so all potential speakers send audio data all of the time.

The situation for virtual world (application) data is potentially more complicated. It is not readily possible to distinguish the traffic from the Host (it appears within the multicast traffic from the world as a whole). It is, however, possible to identify the application traffic for each captain and inhabitant. For the broadcast part of the show (from time 2394 to time 5024) we find the following (average) levels of traffic:

<i>Type</i>	<i>bytes</i>	<i>%</i>	<i>rate bit/s</i>
CaptainA	2560924	0.8	7587.9
CaptainB	2474900	0.8	7333.0
Host/world	1002292	0.3	2969.8
Inhab4A	802916	0.3	2379.0
Inhab1B	677232	0.2	2006.6
Inhab3B	631068	0.2	1869.8
Inhab3A	599076	0.2	1775.0
Inhab2B	532292	0.2	1577.2
Inhab2A	481068	0.2	1425.4
Inhab1A	420252	0.1	1245.2
Inhab4B	415292	0.1	1230.5
Camera2	157756	0.1	467.4
Camera4	157680	0.1	467.2
Camera3	98268	0.0	291.2
Camera1	84616	0.0	250.7
Camera5	80308	0.0	237.9

We observe that:

- The team captains generate around 7500 bits/second on average, over six times the average for the inhabitants. This is as we might expect, since each captain's embodiment comprises four components (head, body, left hand, right hand), and being tracked they are likely to move more (as far as the system is concerned).
- The combined host and world traffic is still only a little more than a single inhabitant's traffic.
- The inhabitants have an average bandwidth (in this show) of around 1800 bits/second. There is considerable variation between inhabitants, from 1230 bits/second to 2379 bits/second, a range of 1149 bits/second. This comprises movement updates and the appearance and disappearance of their speech balloon. Inter-user variations in activity are explored in more detail later in this analysis.
- The cameras (four used for the broadcast plus one additional observation machine for diagnostics and logging) have an average bandwidth of only 340 bits/second (maximum for one camera of 470 bits/second). Even the larger of these is almost four times less than an average inhabitant; the average is over five times less. This figure is lower because the network update rate of cameras was reduced significantly compared to inhabitants. Consequently, the bandwidth requirement approaches that required to just signal that the process is still alive. The system might be further developed to support a purely observing form of participation; in this case the cameras would generate no additional multicast network bandwidth.

From the above we can predict an average bandwidth requirement for each level of participation, including all media:

<i>Level</i>	<i>Bits/s</i>	<i>Components</i>
Host	125000	audio, video, application
Captain	80300	audio, movement (4-part, tracked)
Inhabitant	74600	audio, movement (1-part, joystick)
Camera	340	keep-alive, degraded movement

Clearly, for a finite bandwidth, the level of participation of a user can have a profound effect. In particular, it should (in principle) be possible to add many more camera-like disembodied observers than active participants.

There are three qualifications to make here:

- These figures are without silence detection (which was the case for all four shows), however another situation which allowed its use could significantly reduce the impact of using audio (in the average case, but see also the next point).
- The application/movement figures depend on the actual amount of movement occurring, and particular users, application or interfaces could give rise to much higher levels of updates - see the further consideration of this kind of issue in the later section on user activity modeling.
- Adding any observer (e.g. a camera) may not add much or anything directly to the multicast traffic, but it does add extra load to the multicast reliability mechanisms: additional traffic (in this case unicast UDP) would be generated by those users when multicast packets were lost. This must be taken into consideration when determining how many observing processes might be supported.

4.3 User activity: movement rates

We now move from direct consideration of network traffic to consider user activity and in particular the amount of movement and speaking undertaken by users during the four shows. In this and the following sections we consider the overall rates of activity across all participants and shows and also seek to break this analysis down to explore possible systematic variations due to levels of participation, team, show and phase.

We begin by considering participant movement. Each participant (host, inhabitant and team captain) has an embodiment within the virtual world. We begin this user activity analysis by considering the movement of each user's embodiment. From a network perspective this is interesting because movement update messages need only be sent when an embodiment is moving; no traffic is needed while it is stationary. In OOTW an embodiment can move because:

- the user whom it represents is causing to move, e.g. an inhabitant moving their joystick or a team captain moving a tracker;
- the management system may be imposing (changing) movement constraints on the embodiment which cause it to move; or
- there may be noise in the system (e.g. the team captain's magnetic trackers) which cause the system to see apparent movement.

4.3.1 Overall movement rates

For all participants in all shows we find that the mean level of apparent movement is 59.7% (SD 25.4%), i.e. on average each participant spends 59.7% apparently moving as viewed by the system (and generating update events). This is much higher rate than we have found in previous trials such as ITW (19.6%), COVEN dVS (15.6) or COVEN DIVE (26.3%). Figure 10 shows the cumulative distribution of movement rates for all participants in all shows.

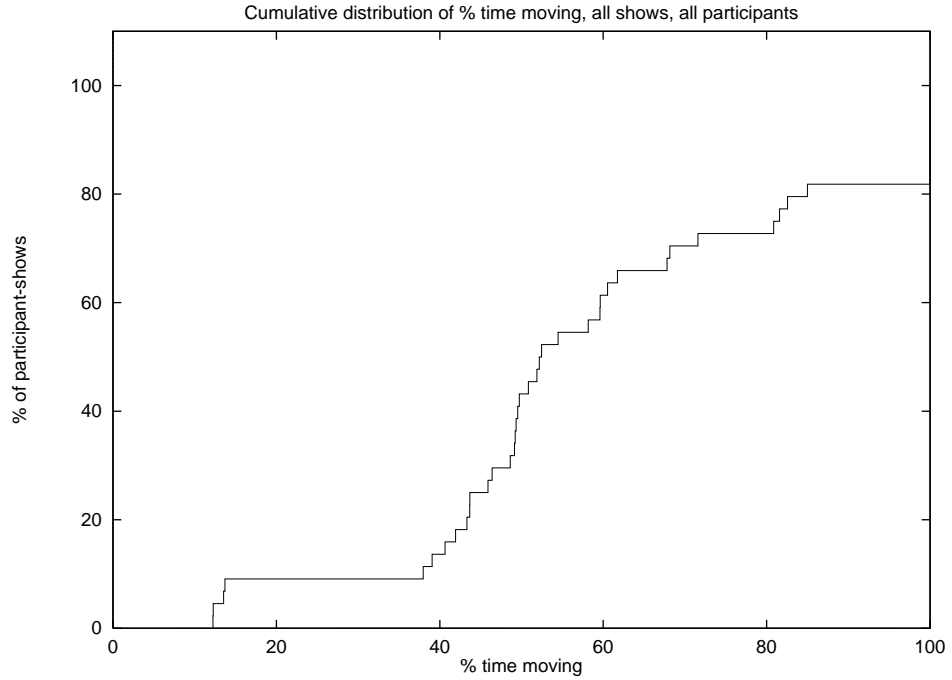


Figure 10 cumulative distribution of movement rates (all-rawmove.cum.eps)

It is apparent from the figure that the distribution is far from normal, with four participants giving very low movement rates (around 13%), and eight giving maximal movement rates (100%). There is a more reasonable spread of values in the range 38-85%.

4.3.2 Movement and levels of participation

To understand this distribution better we show the distributions for host, team captains and inhabitants separately in figure 11.

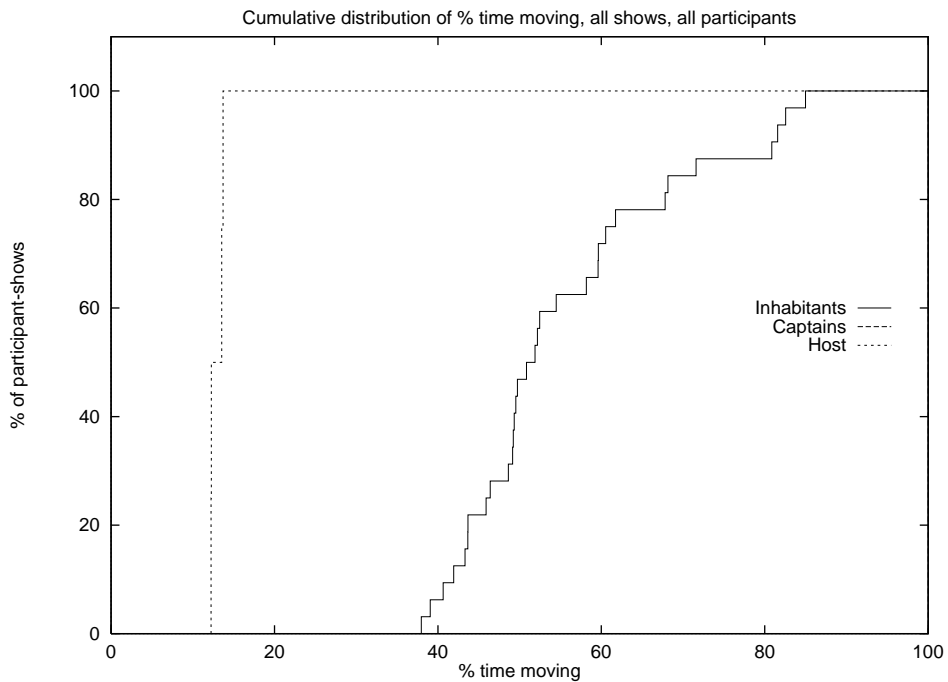


Figure 11 cumulative distributions of movement rates by level of participation (levels-rawmove.cum.eps)

We find that the distinction between levels of participation account directly for the tri-modal form of this distribution:

- The host has a consistently low movement rate, average 12.9%. This is because the host is not directly controlled in the way that the other embodiments are, rather it is moved and positioned solely under the control of the event management system, to position it for different phases of the shows.
- The team captains have consistently maximal movement rates, all 100%. There is sufficient noise in the magnetic tracking system (as used) that the team captains always appear to be moving, as viewed by the system, even if they are "standing still". Consequently the team captain generate position update events continually, throughout the event.
- The inhabitants account for the central region of the distribution, with a mean of 55.5%, standard deviation of 13.2% and a range of 38% to 85%. This is still much higher than observed in previous trials, and the reasons for this are considered next.

4.3.3 Movement and the management interface

Inhabitants' embodiments can be moving for one of two reasons:

- the user is moving the joystick; or
- the event management system is moving them though the movement constraints mechanism.

In order to differentiate these two effects we have extended the user interface and movement messages to distinguish between movement events caused by the user, and movement events caused by constraints. This distinction is only useful for the inhabitants: the host is exclusively moved by constraints while the team captains always appear to be moving themselves (due to variations in tracker positions). For the inhabitants, if we ignore movement caused only by constraints, then we find that the mean rate of movement is 44.5%, with a standard deviation of 15.1% and a range of 24.2% to 82.2%. The cumulative distribution is shown in figure 12, with the original distribution shown for comparison.

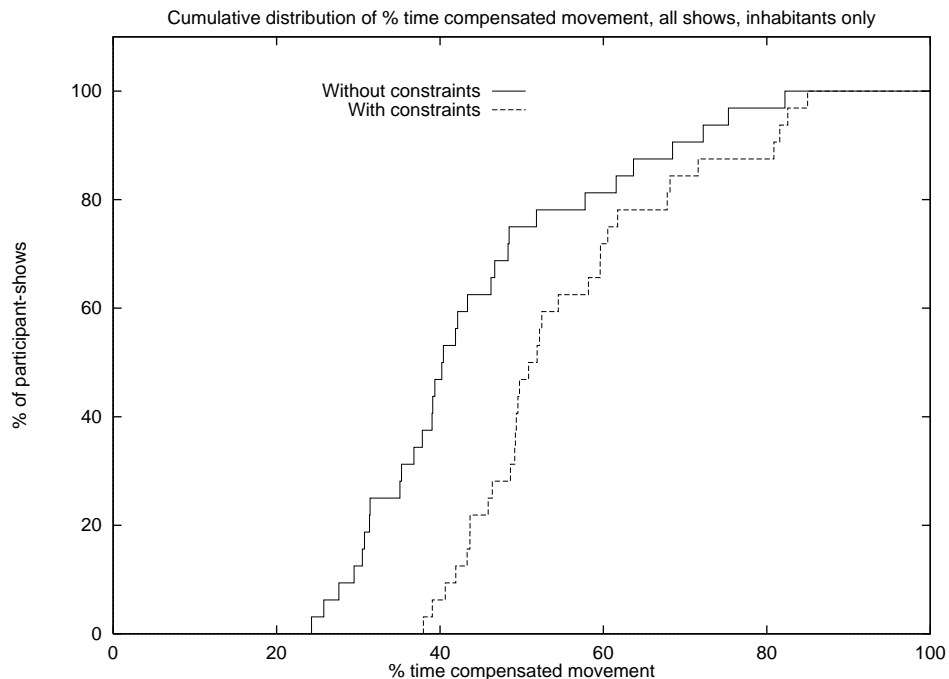


Figure 12 cumulative distributions of movement rates for inhabitants, ignoring movement due to constraints (all-move.cum.eps)

So, the imposition of constraints has produced on average 10% additional apparent movement, and corresponding network traffic. This influence ranges from 13.8% additional movement in the case of the least active inhabitant and only 2.8% in the case of the most active inhabitant.

This compensated level of movement should now reflect the actual actions of the users themselves. In comparing this to the much lower levels from previous trials we identify the following differences:

- The most important distinction is the kind of task being undertaken within the virtual environment. The previous three sets of trials were all essential tele-conferencing applications, with some additional themed content. In OOTW, however, movement is a critical part of all of the games.
- The inhabitants in OOTW had access to no other applications or input devices, and so had no alternative activity within the computer system other than engaging in the show. They did this through moving and speaking. The previous trials had more complex interfaces and the potential for additional parallel activities (e.g. in other applications on the user's machine), in addition to activity within the virtual world.
- The interface for inhabitants in OOTW is a single joystick (and headset for audio). The previous trials used primarily mouse-based interfaces, with some use of the keyboard. The use of the joystick may make movement easier for longer periods.

Further research would be required to explore the relative importance of these possible influences.

4.3.4 Movement and teams

The games had two teams, team A normally had female members, while team B normally had male members (there were one or two instances of team swapping). If we compare the movement rates for the two teams (excluding movement solely due to constraints) we find:

- team A (female) had a mean movement rate over all shows of 40.2%, standard deviation 14.0% and range of 24.3% to 82.2%;
- team B (male) had a mean movement rate of 48.8%, standard deviation of 15.3% and range of 27.6% to 75.3%.

The cumulative distributions are shown in figure 13.

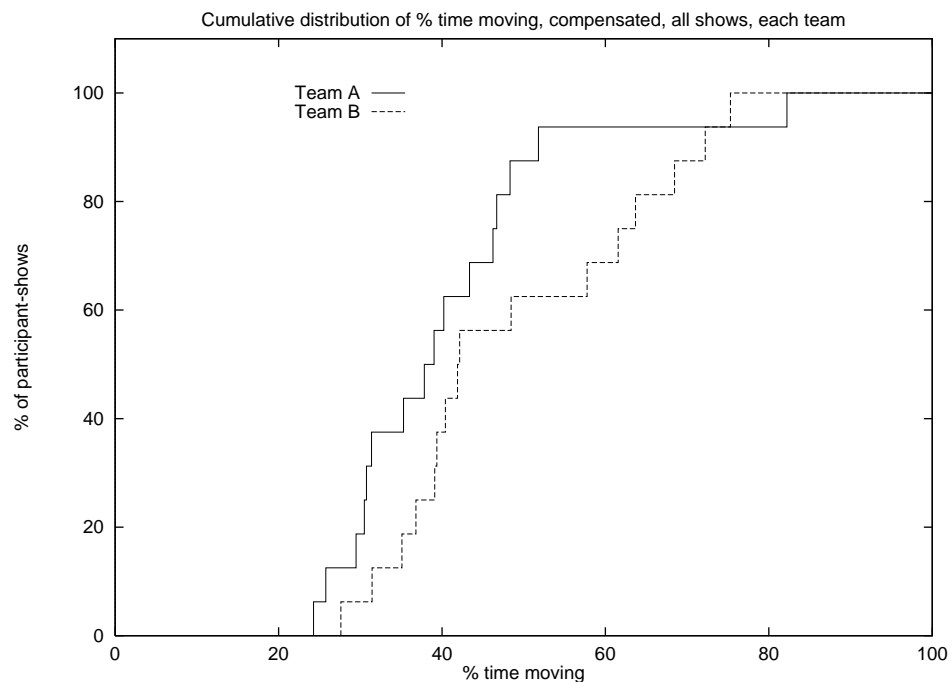


Figure 13 cumulative distributions of inhabitants' movement rates, by teams. (team-move.cum.eps)

This difference is not (quite) significant at the 90% level.

4.3.5 Movement and shows

If we compare the movement rates for inhabitants in each show (again compensating for constraint-driven movement) we find the following per-show figures:

Show	Mean	SD	min	max
1	45.7%	15.8%	31.4%	72.2%
2	44.2%	18.7%	25.8%	82.2%
3	41.4%	11.7%	24.3%	61.5%
4	46.8%	15.8%	30.5%	75.3%

The mean is reasonably consistent for all shows. There is more difference in the variability which could easily be caused by the small number of samples (inhabitants) and outlying data points (particularly active or inactive individuals).

4.3.6 Correlation of movement

In previous studies we have also examined the extent to which a group of participants coordinate their movements, i.e. moving independently or simultaneously. Our previous observations have been that there is slightly more concerted movement than chance, although the overall distribution is still quite close to independent movement. Figure 14 shows the distribution of the number of inhabitants moving simultaneously for all shows. Team captains are ignored because they move "all the time", and the host is moved solely under system control.

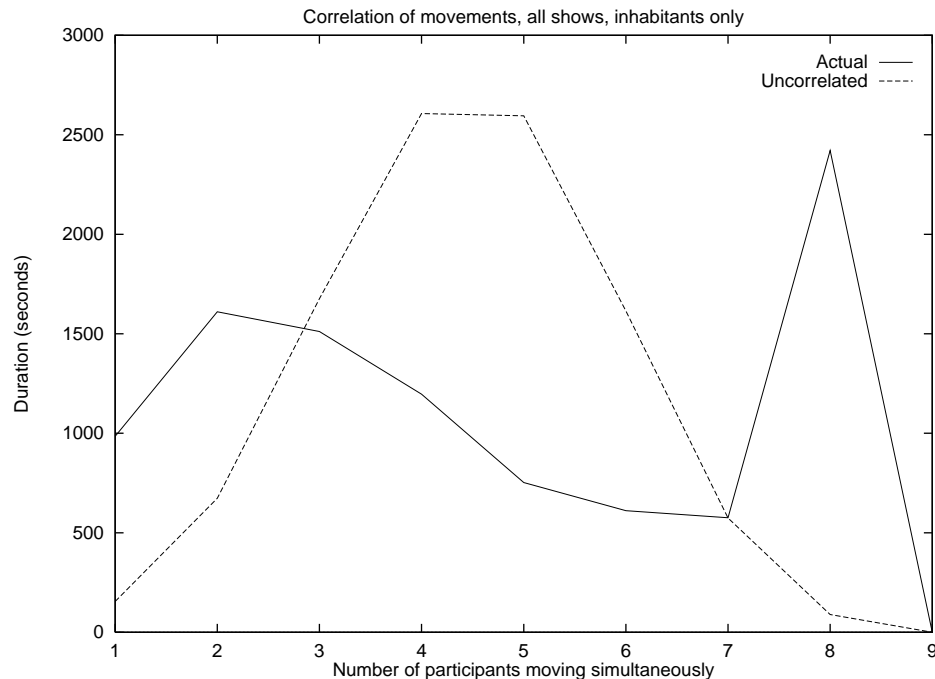


Figure 14 correlation of all movements for all inhabitants. (inhab-rawmove.corr.eps)

The observed distribution is radically different from the uncorrelated curve, in particular there is (relatively) a huge amount of time when all eight inhabitants are moving simultaneously (about 10 minutes per show).

We have already noted that some movement is due to the control of the management toolkit, while some is due to the inhabitants themselves. To assess the balance of these effects figure 15 shows the equivalent distribution for compensated movements, i.e. ignoring movements which are due solely to the application of constraints.

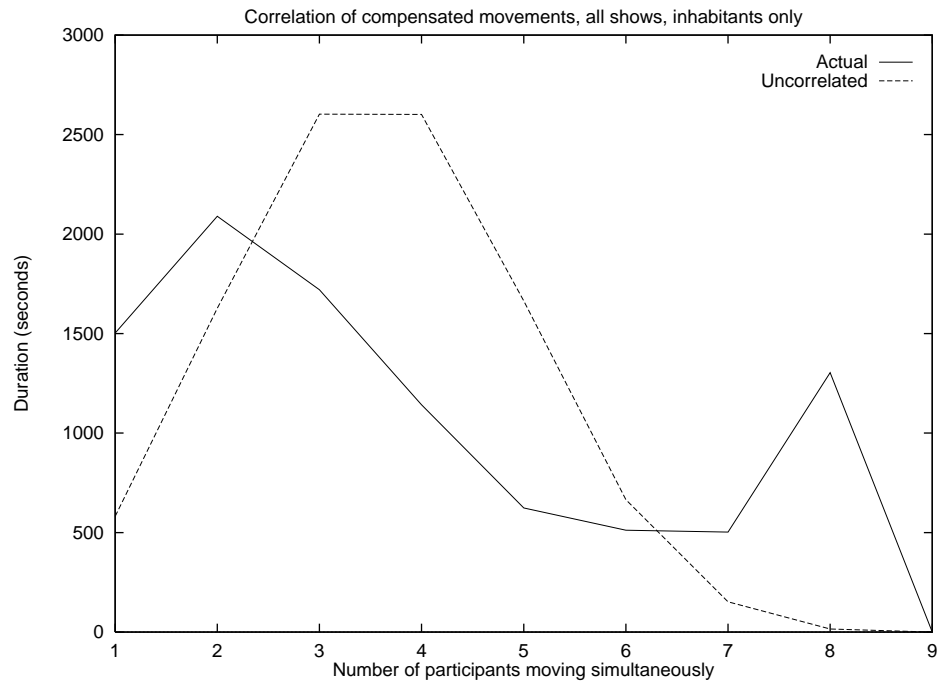


Figure 15 correlation of movements for all inhabitants, ignoring constrained movements. (all-move.corr.eps)

We note that:

- the distribution retains much of its character, with a (relatively) large peak of fully coordinated movement;
- the amount of fully coordinated movement has fallen by about 4.5 minutes per show to around 5.5 minutes per show.

This distribution is still very different from the distributions observed in previous tele-conferencing type trials. The much greater correlation of movement is almost certainly primarily due to the structure and character of the games. For example, the space frog game involves all of the inhabitants running about after space frogs, and the race involves all of the inhabitants pulling the jet car to the finishing line. Clearly, the system and network must be provisioned in anticipation of significant periods with every participant moving. This bears out the reservations which we have expressed in previous analyses concerning the likely task-dependency of measures such as this.

The additional coordinated movement is largely due the transitions between arenas on the travellers when all of the participants are moving at the same time. All of the inhabitants are also moved about at the end of each game, when they are gathered together at the exit of the arena.

4.3.7 Movement and phases

In section 4.1.1.1 we noted that there was a link between application multicast traffic and event phase structure. In this section we look at the relationship between phases and movement rates which will be one component of this relationship.

First, we consider the overall (network-visible) movement rates, due to both participant actions and management constraints. Figure 16 focuses on a small section of show2, including the introductory football match and the space frogs game. It shows the average number of participants moving and the maximum number of participants moving in each five second block of the period in question.

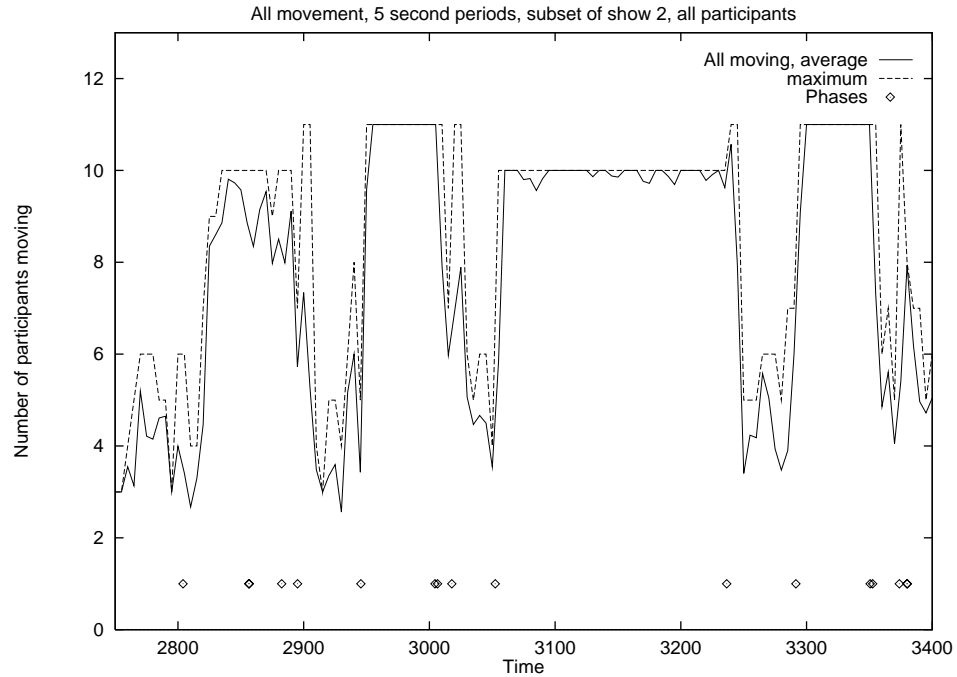


Figure 16 all movement activity, average and maximum number of #participants moving, for a subset of show2. (show2-rawmove-5.eps)

We note that:

- All 11 participants (including the host) are moving during the transitions between arenas (2945-3020 and 3290-3370) and at the start of each entry/exit phase, when the host and all participants are positioned at the entrance/exit of an arena (times 2900, 3020, 3240 and 3370).
- We already know that this view represents the team captains as moving all of the time, giving an absolute minimum of 2 participants moving, even at the quietest time. In fact, the average level never quite gets this low (e.g. during the introduction before the football match, time 2750-2820, or at the end of space frogs game, 3240-3290).
- All eight inhabitants are also moving (generally) during the football game (time 2820-2890) and the space frog game (time 3050-3240).

From the above the effect of dynamic constraints (i.e. the travellers) is clear.

If we ignore constraint-related movement and visualize the same information for the inhabitants deliberate movement we get the view in figure 17.

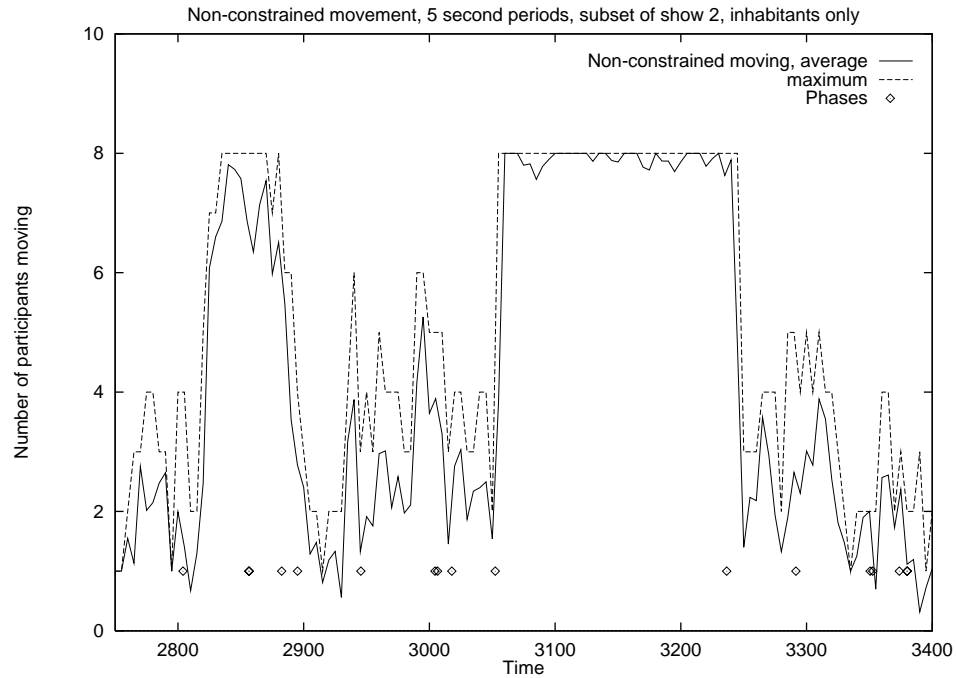


Figure 17 non-constraint movement, average and maximum number of participants moving, for a subset of show2. (show2-move-5.eps)

We see clearly that the inhabitant are almost all moving (deliberately) all of the time during the football match (2820-2890) and the space frog game (time 3050-3240), independent of any constraints. For the remainder of the period in question there is consistently less attempted movement, although the level is quite variable. Even when the inhabitants are "pinned to the spot" (e.g. at the entrance or exit of an arena) they can still look around if they wish to. So, with the exception of the focused games, there are no clear distinctions between the different the inhabitants levels of activities at other phases.

4.4 User activity: audio participation rates

Having considered participant movement in the previous section we now consider audio participation rates. Each participant (host, inhabitant and team captain) has an open microphone which they can use to talk to the other participants. In the case of the inhabitants and the team captains this is a microphone/headphones combination which allows hands-free real-time audio interaction.

4.4.1 Silence detection

The audio communication is provided by a general-purpose audio server, one per participant, which uses packetised multicast peer-to-peer audio data distribution. In the previous trials which we have analyzed a silence detection algorithm has been used by the audio software so that audio packets are only transmitted when sounds are being received by the microphone. This creates a direct link between speaking (or at least the production of noises) and the generation of network audio traffic. It was our intention to use the same thing in OOTW. However, when sound-checking the system in rehearsal it was decided that the silence suppression mechanism should be disabled. This was done because the sound-scape was already quite chaotic (eleven potential speakers plus sound effects) and the silence suppression was making things more difficult (e.g. losing the beginnings of quiet phrases or cutting off altogether in very quiet utterances).

Consequently, from the perspective of the network it was as if every participant were speaking all of the time! This accounts in part for the fact that audio traffic dominates the total network bandwidth, e.g. 91% of the traffic in show 2 (see section 1.1 and figure 2). In the circumstances, it is fortunate that we did not assume any lower level of audio activity in provisioning our network and systems.

Whilst the network's view of audio traffic was rather degenerate in OOTW we would also like to explore what would have happened if silence suppression had been retained. This will allow us to apply our experiences in OOTW to other settings in which silence suppression is used. It will also allows us,

to a first approximation, to reason about how much participants actually spoke, and to explore whether this was influenced by the various distinctive elements of OOTW such as levels of participation and phase structure.

For this post-event analysis we have approximated the operation of the system's normal silence suppression algorithm by applying it to the 56 bytes of audio sample data from each audio packet which are preserved in the network traffic log (captured by tcpdump). This is 17.5% of the total audio data on which the algorithm would normally operate, and so it is likely to underestimate the amount of speaking that would have been detected had silence suppression been used in the event. Having said this, it should give quite a good approximation for our purposes.

4.4.2 Overall audio rates

For all participants in all shows we find that the mean level of apparent audio participation is 40.7%, standard deviation 21.8% and range 9.6% to 81.1%. This value is also higher than we have found in previous trials such as ITW (26.4%), COVEN dVS (5.2%) or COVEN DIVE (8.1%). Figure 18 shows the cumulative distributions of audio rates for all participants in all shows.

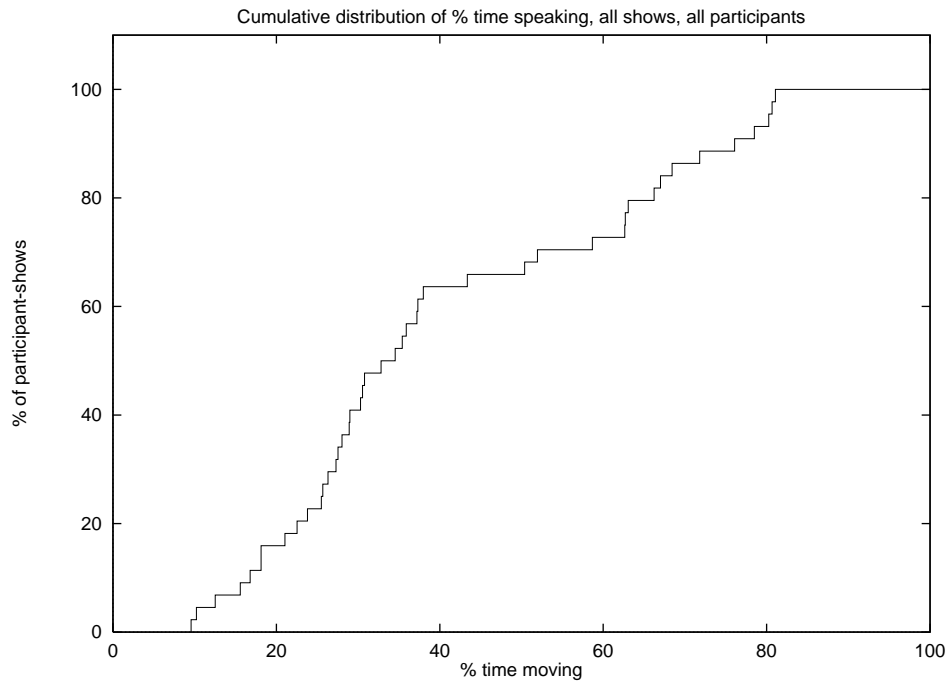


Figure 18 cumulative distribution of audio participation rates for all participants. (all-audio.cum.eps)

As with movement, the distribution is clearly not normal in form.

4.4.3 Audio and levels of participation

Figure 19 shows the distribution of audio rates broken down by level of participation.

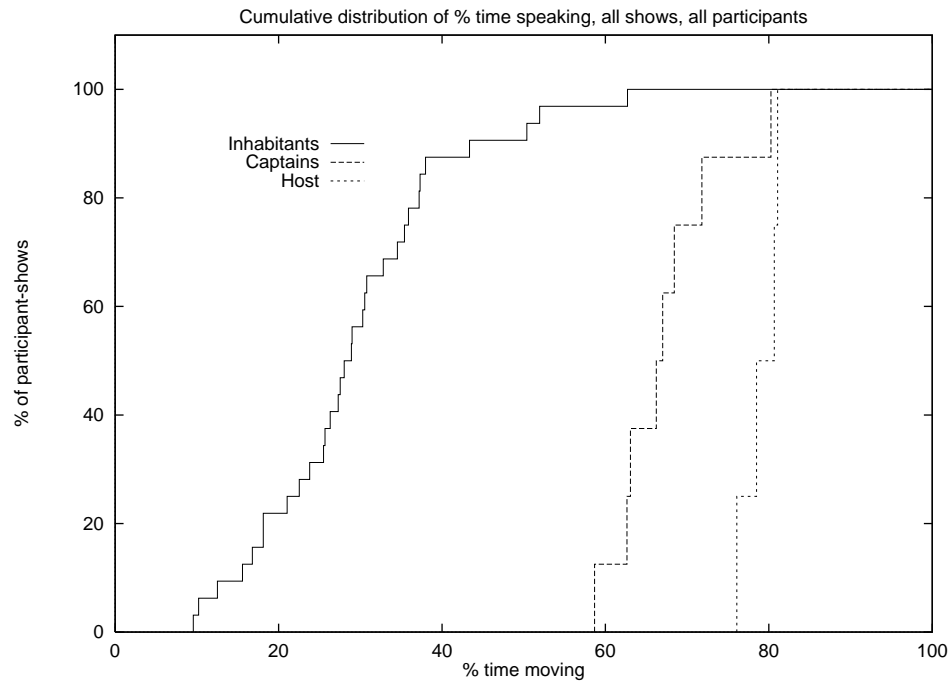


Figure 19 cumulative distribution of audio rates by level of participation. (levels-audio.cum.eps)

- The host has a consistently high apparent audio rate, average 79.1%, range 76.1% to 81.1%. We believe that only part of this is due to the actual speech of the host; the host was placed in the lighting booth overlooking the theatre and sound from that space (e.g. the audience and the PA) leaks back into the host's microphone.
- The team captains have a higher-than-average audio rate of mean 67.3%, standard deviation 6.6%, range 58.7% to 80.2%. This higher rate may be due to a combination of speaking more - they have clear roles and responsibilities in the event - and sound pickup from the PA - they were standing at the front of the theatre space.
- The inhabitants have a mean audio rate of 29.3%, standard deviation 11.9%, range 9.6% to 62.7%. This is directly comparable with that for the ITW trials of 26.4%, though still much higher than that found in the other trials.

We conjecture that the higher speaking rates for inhabitants may be due largely to the higher pace and more structured roles and interaction in OOTW compared to the previous trials in which we have been involved. These have tended to be rather pedestrian and disorganized, with no clear patterns of responsibility and less focused involvement.

4.4.4 Audio and teams

If we now compare the audio rates for the two teams we find:

- team A (female) had a mean audio rate of 27.6%, standard deviation 11.4% and range of 9.6% to 50.4%;
- team B (male) had a mean audio rate of 31.0%, standard deviation of 12.6% and range of 15.6% to 62.7%.

The cumulative distributions are shown in figure 20.

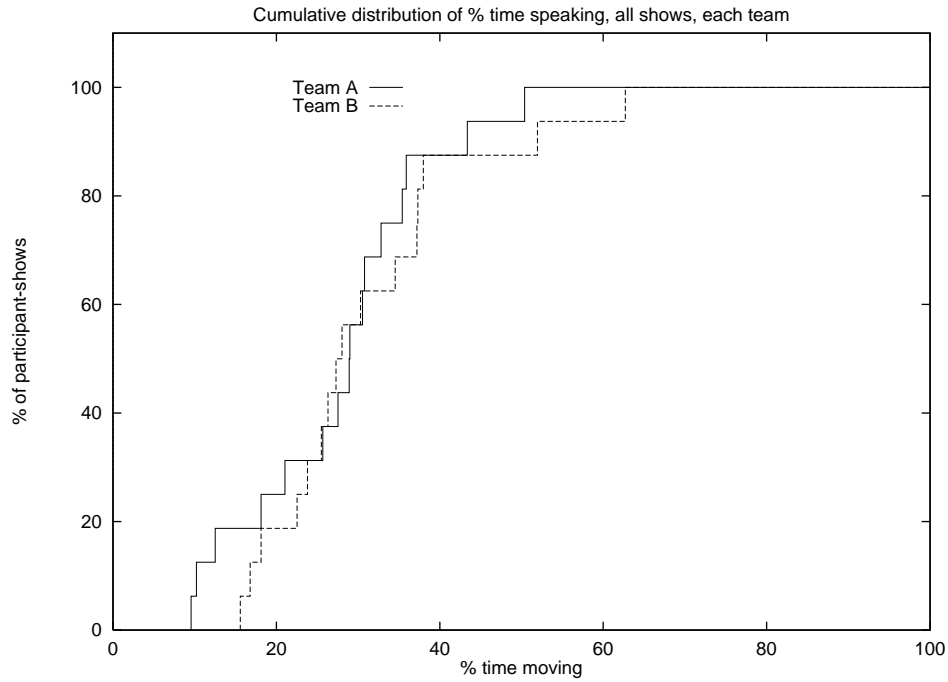


Figure 20 cumulative distributions of inhabitants' audio rates, by teams. (team-audio.cum.eps)

The difference is not significant.

4.4.5 Audio and shows

If we compare the audio rates for all inhabitants in each show we find the following per-show figures:

Show	Mean	SD	min	max
1	28.9%	9.1%	18.1%	43.4%
2	36.8%	15.5%	9.6%	62.7%
3	22.9%	9.4%	10.2%	37.2%
4	28.6%	10.4%	16.8%	51.9%

Audio rates are much more variable between shows than movement rates. However, without additional information there are no clear inferences to be drawn from this variability, other than to note its existence.

4.4.6 Correlation of audio

As with movement we also wish to consider the extent to which a group of participants coordinate their movements, i.e. moving independently or simultaneously. Our previous observations have been that there is slightly less concerted speaking than chance, although the overall distribution is still quite close to independent movement (extremely close in the case of the COVEN DIVE analysis). Figure 21 shows the distribution of the number of inhabitants "speaking" simultaneously for all shows. The values are also shown in the table below. We restrict our consideration to inhabitants because of the different characteristics of the other participants, especially the host, which might affect the outcome.

<i>Number</i>	<i>Actual</i>	<i>Chance</i>
1	2551	2064
2	2369	2999
3	1550	2491
4	904	1293
5	558	429
6	392	89
7	243	10
8	133	0.5

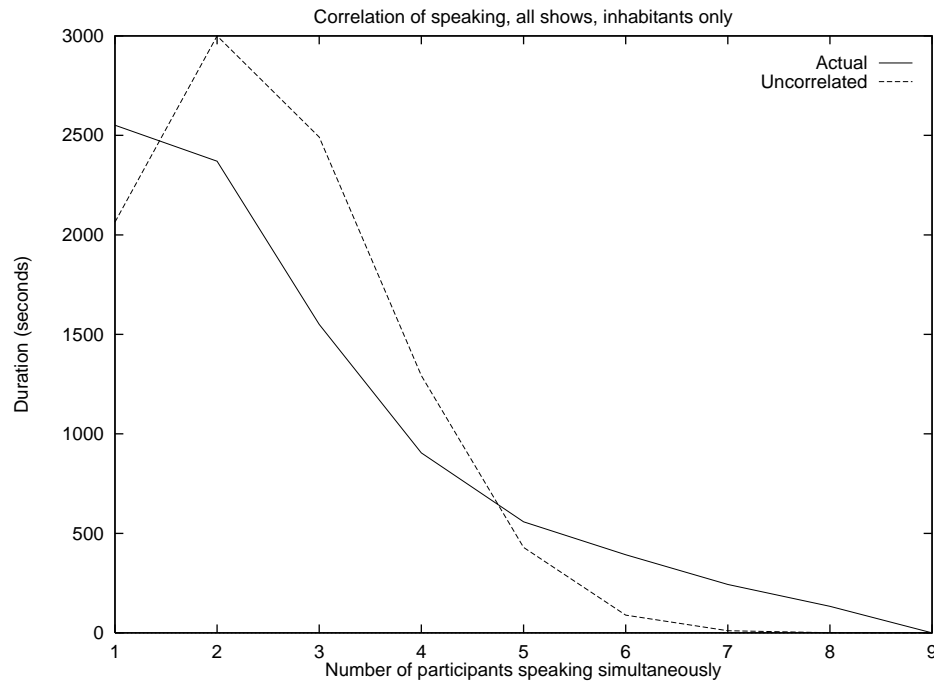


Figure 21 correlation of audio for all inhabitants. (*inhab-audio.corr.eps*)

The deviation from chance is much less than for movement, but still considerable. For example, one might discount the likelihood of all inhabitants speaking simultaneously based on the random model (a total of half a second over all shows). In fact, over two minutes of all eight inhabitants speaking simultaneously is observed.

We have conjectured in previous analyses that significant bursts of correlated audio activity could arise in worlds with common elements of interest and we believe that this is borne out by this analysis. Event elements such as team dynamics (e.g. cheering/heckling), response to game events (e.g. scoring, approaching the finish) and coordinated phases (e.g. being released to start a game) might all tend to produce coordinated vocal activity.

It is not possible in this analysis to rule out the possibility of external interference. For example, loud noises in the environment might be picked by all inhabitants' microphones. A much more controlled study and isolated environments would be needed to assess this possibility. However, we note that such external influences could be significant in real situations as well, for example if an on-line virtual event is linked to a television broadcast, then that would serve as a possible global source of coordinated interference. In any case, we believe that the possibility of coordinated audio activity must be considered very carefully if the quality of the experience is to be assured.

4.4.7 Audio and phases

In section 4.3.7 we looked briefly at the relationship between phase and movement. In this section we view (estimated) audio activity for the same portion of show 2. As in figure 16, figure 22 focuses on a small section of show2 including the introductory football match and the space frogs game. It shows the average number of participants "speaking" and the maximum number of participants speaking in each five second block of the period in question.

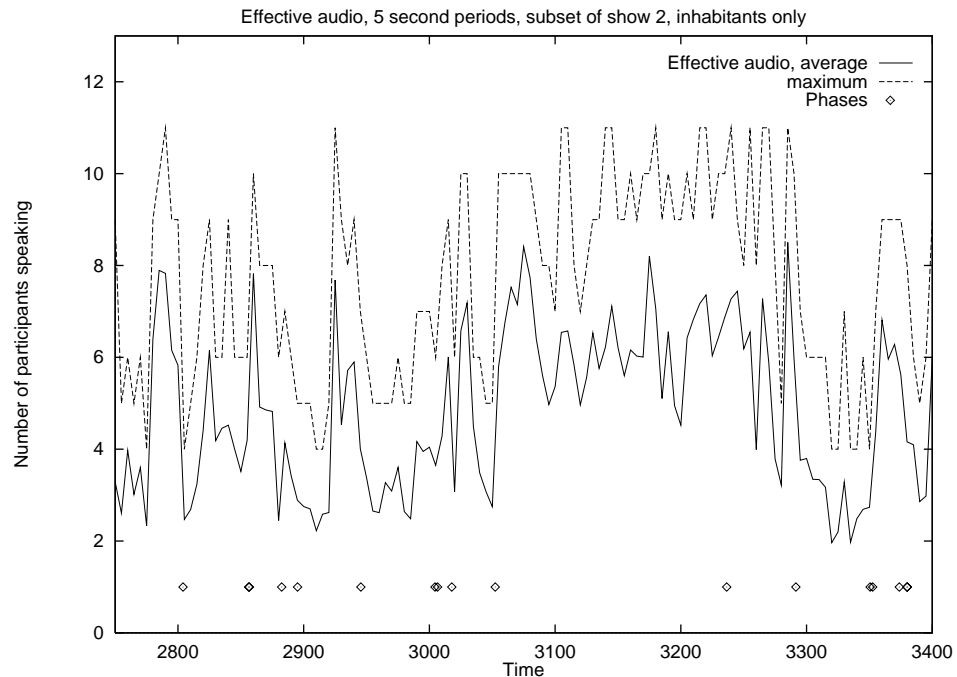


Figure 22 audio activity, average and maximum number of participants "speaking", for a subset of show2. (show2-audio-5.eps)

Comparing figure 22 to figure 16 or 17 we see that the audio activity is much more variable and less clearly linked to phase structure. However, there are a number of observations which can be made:

- The largest consistent period of audio activity occurs during the space frog game (time 3050-3240) when (from observing the video we see that) everyone is shouting, giving advice, cheering, etc. There are many occasions in this period when all participants (and the host as well) are active in the audio medium.
- This audio activity persists after the end of space frog game into the post-game debriefing session; it only subsides then the participants begin the transition to arena 2 at time 3300. The participants are also relatively inactive during the transition to arena 1 (time 2945-3020), when the host is delivering a monologue introduction the space frog game.
- We can tentatively identify smaller bursts in audio activity before and after the football match (around times 2860 and 2430), on arrival at arena 1 (time 3020) and on arrival at arena 2 (around time 3360). These correspond to particularly vocally active periods in the introduction/debrief process at each arena.
- Based on only event phase it is impossible to identify times when large amounts of speaking would definitely not occur

4.5 Summary

This completes our analysis of the network and user activity records from the Out Of This World. We highlight the following observations in particular:

- The overwhelming majority (98.2%) of network traffic was multicast. Of this, packet audio accounted for 91%, video for 4% and the virtual world/graphics for 3.2%.

- In part, the large amount of multicast audio data was due to the disabling of silence suppression (in order to increase the quality and intelligibility of the audio mix); without this the average audio bandwidth might have been reduced by about 60%.
- There are clear differences in the network traffic generated by users at different levels of participation. In particular, the host generates the most traffic (since it includes video), the captains next (more complicated multi-part embodiments) and then the inhabitants. The cameras generate very little network traffic (around 0.5% of an inhabitant) since they cannot speak and are not visible in the world. Consequently, the division of participants between these levels can have a large impact on the overall network requirements.
- Potentially many more participants could be introduced at the level of virtual cameras, i.e. able to freely navigate the scene, selecting their own viewpoint on the event. In principle these might generate no additional network traffic. However, careful consideration would have to be given to their potential impact on the multicast reliability mechanisms.
- With regard to user movement, we find that levels of participation are again an important predictor for movement rates. Specifically, the team captains move all of the time (because of the use of magnetic position trackers) whereas the host moves only sporadically, under the control of the management interface. This is directly attributable to the form of interface chosen (or required) for each level of participation.
- The inhabitants move much more than in previous trials we have analyzed. Only part of this is due to the operation of the management interface, which moves inhabitants about at particular phases of the event (e.g. on the traveller). Compensating for this we find that inhabitants are still moving 44% of the time. We conjecture that this higher rate of movement is due to a combination of factors, including: the more active nature of the "show" compared to previous tele-conferencing style applications, a simpler and more direct interface (joystick only) and the absence of alternative activities (no keyboard, no access to other applications).
- The use of movement constraints produces periods (about 5 minutes per show) when all of the participants are moving at the same time. This has potential implications for network and client resource provisioning, since they must be able to cope with maximum levels of activity.
- Furthermore, ignoring the effects of constraints, the inhabitants still move all together a significant amount of the time (again about 5 minutes per show). This is completely different to the patterns of movement observed in the previous tele-conferencing style trials, when movement by different participants was only weakly correlated. We believe that this is due to the structured nature of the event and to the tasks which it comprises, e.g. the race requires coordinated movement from all inhabitants.
- There is a visible link between phases of the show and levels of movement. Inhabitants (voluntarily) move a great deal during the more active games. All participants (involuntarily) move during links and transitions between arenas, under the control of the management interface. At other times a greater variability of movement rates is observed.
- With regard to audio we note that silence suppression was avoided in this event for reasons of perceived audio quality and clarity of interaction. This produced higher average network bandwidth requirements (though it did not affect peak requirements, as noted below).
- By simulating the effects of silence suppression from packet data recorded in network log files we were able to explore the characteristics of audio activity in these trials.
- As with movement, we found a link between levels of participation and audio activity. The host was the most active "speaker", although we suspect that this includes the effects of audio spill from the performance space. The team captains were active in audio much more than the inhabitants, though again there is the danger of audio spill from the house PA increasing the apparent level of speaking from the team captains.
- The inhabitants were classified as "speaking" more than in previous analyses, although the results were quite close to those for the ITW trials. The high level in ITW trials was due, at least in part, to less effective silence detection and to the use of open speakers rather than headphones by at least one participant. We suggest that this relatively high level of audio activity (mean 29.3%) reflects the greater pace and structured interaction of this event compared to previous trials.
- Previous analyses have found very little correlation between different speakers, and in some cases negative correlation (i.e. that speakers avoid speaking over one another to some extent). However,

in this analysis we found a dramatic increase in participants speaking at the same time. For example, there were two minutes during which all eight participants were considered to be simultaneously audio-active, compared with an (uncorrelated) expectation of 0.5 seconds. This casts doubt on the reliability of silence suppression as a means of reducing peak system requirements for audio bandwidth and processing, at least for applications of this nature.

- There is some apparent linkage between show phase and audio activity, although it is more limited than that observed for movement. Most periods when all participants are speaking occur during the active games, however there are other periods of great audio activity during the more static introductory and debriefing sessions.

4.6 Conclusions from OOTW

Altogether, we believe that the analysis of Out Of This World has demonstrated very clearly that patterns of audio activity and movement are profoundly dependent on the applications and interfaces used. In particular, a highly structured event such as this one is liable to produce highly correlated bursts of intense activity, which mean that any notion of design or provisioning based on average levels of activity must be treated with extreme caution. For systems which anticipate high levels of concurrent activity we suggest (for MNIT) the exploration of the following approaches to enhance scalability and consistency of performance and requirements:

- Simple peer-to-peer audio, even with silence suppression, becomes increasingly demanding where rates of participation and correlation of activity are high. The alternatives are to adopt a relatively rigid form of floor control (which may be suitable in some applications) or to introduce server audio components which can mix multiple audio streams "in the network". Such an approach could still use multicasting to distribute the final composite audio channel, but the variability in network and processing requirements could at least be localized to those (potentially distributed) server components. There will also be necessary sacrifices in using such an approach, such as loss of per-speaker separation, increased latency. But this may be the only acceptable solution. An example of such an approach is the crowd aggregations provided in MASSIVE-2, which mix the audio of their members and re-broadcast a composite audio stream to the rest of the virtual environment.
- Many current CVE systems have per-user flow control limits, but no overall (per world, per region, etc.) flow control limits. For example, in MASSIVE-2 each user sends rate-limited updates when they move, but this rate limit is independent of the other activity in the world. To cope with higher levels of participation and especially with variable levels of participation or activity these relatively elastic communication activities could be constrained to adapt to the current overall level of activity within the system. For example, if everyone is moving at the same time then the individual rate of movement updates is reduced accordingly. Similarly, if increasing numbers of participants start to introduce video streams into the world (e.g. video faces) then the bandwidth allocation of each reduces accordingly. Consequently the overall world activity remains bounded, even in the face of unpredictable and highly correlated bursts of user activity. This is analogous to (though potentially more complicated than) the rate control mechanism used for RTCP (Real Time Control Protocol) packets, which effectively share out a collective bandwidth allocation between all current session members.

We believe that the combination of these two approaches, abstraction services and adaptive clients, will allow the construction of more scalable CVEs which provide more graceful response under variable patterns of load (e.g. variable user activity) than do current systems.

5 Conclusions

A significant body of resources and experience has been built up, including:

- Experimental methodology and issues.
- The dummynet network simulator.
- Traffic flow analysis tools.
- User activity analysis tools.

These have been applied to the Out Of This World experimental event, yielding significant information and insights about network requirements and user activity in Inhabited Television (as distinct from previous experiences with, for example, teleconferencing). These are also a sound basis on which to build, when assessing any future experiments and trials.

Although the strategic decision has been made to suspend plans for trials on the Future's Testbed, this experimental work package has still yielded significant – and generalisable – results.

6 Appendix A: Porting and installation of MASSIVE-2

At the outset of the project, the intention was to install and use MASSIVE-2 on the BT Future's Testbed as the experimental platform. To this end the following porting and installation work was carried out.

At the formal start of the project (and in the run up to it) MASSIVE-2 was ported from SGs to PCs (running under NT 4). Video handling was also ported to a non-platform specific video coding (JPEG using a publicly available compression library). Support for Java integration (i.e. calling between Java objects and MASSIVE-2) was also added during the port.

Over the summer of 1997 Kostos (BT) installed earlier versions of MASSIVE-2 at BT, primarily on SG machines. Hooks for controlling RSVP were added to MASSIVE-2 at this time, although problems with operating system and router support for RSVP meant that they were never used successfully.

Chris Greenhalgh visited BT on 7th January 1998 to sort out final problems with the installation on PCs at BT. MASSIVE-2 was successfully installed and configured, and a brief wide-area trial was held between Nottingham (Dave Snowdon) and BT (Chris Greenhalgh and Ian Fairman); this was successful.

7 Appendix B: Analysis tools documentation

This appendix describes: network packet analysis tools; flow analysis tools; and user analysis tools.

7.1 Network analysis tools – packets

These tools support operations on tcpdump packet logs, including conversion to fs format for input to flow analysis, extraction of individual fields from packets, and comparison of packets in two dump files in order to establish clock offsets and delays.

7.1.1 Tcpdump2fs

This program converts from tcpdump format file (a subset thereof) to fs format files for use with the flow analysis tools. It will also attempt to spot and eliminate packets which have been recorded multiple times (same packet data including link-layer). I have seen this appear to happen on some machines.

It also (for the COVEN project DIVE analysis) performs optional mappings specific to DIVE (3.2) used with the “DiveBONE” to:

- remap destination IP address for DIVE Multicast packets to distinguish between different kinds of packets, specifically, data, object, req and also what it believes to be duplicate requests, due to reliability mechanisms.
- remap source IP address and port for DIVE Multicast packets to originating IP address when packets have travelled via the DiveBONE (otherwise the local proxy server would be the originating address).

It currently assumes that DIVE multicast messages are sent to UDP port 1166 and that DIVE audio messages are sent to UDP port 3444.

When remapping destination ports to show data types the outputs are:

- DATA_PKT(not group)= 1166
- DATA_PKT(group)=1167
- OBJ_PKT(not entity)=1168
- OBJ_PKT(entity, initial)=1169
- OBJ_PKT(entity, duplicate?)=1170
- OBJ_REQ_PKT(not entity)=1171
- OBJ_REQ_PKT(entity, initial)=1172
- OBJ_REQ_PKT(entity, duplicate?)=1173
- Other=1174

When remapping source addresses for DiveBONE it will DROP all unicast UDP traffic, which is assumed to be inter-proxy traffic!

Usage:

- v : verbose reporting
- D : "ideal dive", ie remap dive destination addresses, as above.
- p : hide proxy, i.e. remap dive multicast source addresses.

Reads tcpdump file from standard input. Outputs to stdout.

NB requires reasonable capture len, about 111 bytes for DIVE proxy remapping and about 95 bytes for dive destination remapping.

7.1.2 Tcpdump2data

This program reads packet records from a tcpdump file and outputs specified bytes/words/long-words from packets, one packet per line in plain ascii.

Usage:

- a <float> : a_time - minimum time of packets to output (unix time).
- b <float> : b_time - maximum time of packets to output (unix time).
- f <rulefile> : filter output so that only packets matched by this rule set are present in the output. Addresses are NOT changed by "!" rules. See flow analysis tools for more information on rule sets and summaries.
- r <rulefile> : apply this rule set to summarise/reliable flow data, as it normally would in flowfilter. Does not actually changed output values. Only used as a precursive to a simpler -f rule set to perform final filtering.
- o <output-description> : specify (additional) output rule, in format "<size>@<offset>:<format>" where <size> must be 1, 2, 3 or 4, offset is from the start of the IP header, and format is a printf format such as "%ld", "lh" or "%lx".

Each line of output begins with the UNIX time in seconds (a float) followed by the output values in the order specified.

7.1.3 Tcpdump_compare

This program finds (by byte pattern) packets sent from host1 to host2 and vice versa.

Usage:

```
Tcpdump_compare [-v] tcpdumpfile1 host1 tcpdumpfile2 host2 [bad-server-ip]
```

tcpdumpfile1 should have been logged on host1, and tcpdumpfile2 logged on host2.

Outputs hosts and time pairs, packet type and size.

If one of the tcpdump machines is recording duplicate entries then that is the bad-server-ip optional argument.

It doesn't always work, especially if many packets are not recorded in tcpdump file(s). It is sensitive to the start times of the files, the first time they exchange messages and the order in which the files are mentioned. Ideally, both log files should start before the machines exchange any data packets.

7.1.4 Find_rtt

Usage:

```
find_rtt <tcp-compare-file-a-to-b> <tcp-compare-file-b-to-a>
```

input: two output files from tcpdump_compare for each direction in a single session.

output: mean and sd of rtt

These files would need to be extracted from tcpdump_compare output by filtering.

7.2 Flow analysis tools

These tools support the extraction of packet flows from fs format packet logs. They also allow summaries of flows to be generated using pattern matching of flow characteristics. The tools are fs2flows, flowfilter, flow2data and summary. Some of the file formats are described first.

7.2.1 File formats – fs, flow

“fs” is a simple IP packet format (used by NLANR) which preserves minimal information on src/dest, protocol, size and time. More compact than tcpdump, but of course loses other application-level header information.

The fs format, from <http://www.nlanr.net/Flowsresearch/>:

"The output to stdout is 24 bytes/packet header:

```

+-----+
0 |          timestamp in seconds          |
+-----+
1 |          timestamp in microseconds     |
+-----+
2 |          IP source address             |
+-----+
3 |          IP destination address        |
+-----+
4 | IPProt | TCPflags|  Packet-length      |
+-----+
5 | destination port |  source port      |
+-----+

```

The flow format currently records both information on completed flows (line beginning 'A') and partial information for current flows (time period records begin 'I' - interval - and flow records begin 'T'). See fs2flows or flowfilter for details.

Main entry types are as listed below.

1) Time interval for following 'T' records, unix time, from, to.

```

I 907160553.000000 907160613.000000
   from-time      to-time

```

2) Information on a flow (or flow summary group) for the current time interval only (last 'I' record).

```

T: 00000000:00000000:000:00000:00000 7   other 6       24   1152
   src-ip  dst-ip  prot  sport  dport  rule  name  #flows  pkts  bytes

```

Note that the rule/name field are filled in if the flow matched a rule filter. The number of the rule in its file and the name are preserved. The #flows field indicates the number of low-level IP flows which are represented here. The packet and byte count are for the current interval, only. Note also that the entry may appear as "T-S:" in some cases to distinguish summary rule use within fs2flows itself.

3) Complete information on a single completed flow.

```

A: 00000000:00000000:006:00000:00000 6   tcp 716 907160603.935520 907167313.414887 228850 21416531
   src-ip  dst-ip  prot  sport  dport  rul  nam  #fl  start-time  end-time  pkts  bytes

```

The fields are essentially the same as a combination of I and T records. The only really new information is the more exact start and end time of the flow (rather than the I period in which it started and ended).

7.2.2 File formats – rules

The rules file format is implemented by the routines in C files summary.h/c which in turn is used in fs2flows, flowfilter and tcpdump2data. flowfilter is the most commonly used of these. Rules provide a combined means to:

- select packets/flows with particular characteristics for inclusion; and
- collapse multiple flows onto a single combined summary flow.

As in fs/flow, flows are identified by the quin:

(src-ip-address, dst-ip-address, ip-protocol, src-port, dst-port)

Where src-port and dst-port are defined for UDP and TCP as the respective transport layer ports. For other protocols they are typically not defined (at least at present).

Rules are supplied in a rule file. A leading '#' indicates a comment line. Each rule is placed on a line of its own, and the grammar for a rule is as follows:

- *rule = name src-rule dst-rule prot-rule sport-rule dport-rule* ['+']

- *src-rule, dst-rule, prot-rule, sport-rule, dport-rule = num-rule*
- *num-rule = pattern ['!']*
- *pattern = '*' | value | mask-rule | range-rule | choice-rule*
- *mask-rule = value '/' mask-len*
- *mask-len = value*
- *range-rule = value '-' value*
- *choice-rule = '{' value (',' value)* '}'*
- *value = num ['!' num ['!' num ['!' num]]]*
- *num = [0-9]+*

Notes:

- A trailing '+' after a rule causes the flow to continue being matched against the following rules. Otherwise matching against this rule file would stop once a match had been found.
- A pattern followed by a '!' causes that field of the flow to be rewritten to the lowest allowed value for the matching rule. This is the basic summarisation mechanism. For example, a '*' rule gives a zero final value for the field, a range rule take the lower bound, etc.
- A '*' pattern matches any value of that field.
- A value rule can be a single decimal number (e.g. for a port or protocol number) or a dotted base-256 number, as in the normal format for IP addresses. Note that 1.0.0 will be hex 0x10000 - no additional trailing zeros are assumed, they must all be entered explicitly. Value is an unsigned 32 bit integer.
- A mask rule matches the specified number of leading bits. E.g. "224.0.0.0/3" matches the first 3 bits against the first three bits of "224.0.0.0" (i.e. 111), which identifies IP class D & E addresses (multicast and experimental).
- A range rule matches all values in the range. E.g. "1-1023" identifies reserved port numbers.
- A choice rule matches any of the alternatives, e.g. "{6,17}" identifies TCP or UDP protocol numbers. A choice is currently limited to 10 entries; multiple rules must be used (e.g. with same lowest value or rule name) to achieve the effect of more alternatives.

Example: this rule file identifies UDP multicast, UDP unicast and TCP traffic. All remaining traffic is grouped under "other".

```
# name  src dst          prot sp dp
MC-UDP *!  224.0.0.0/3!  17  *! *!
UC-UDP *!  *!             17  *! *!
TCP    *!  *!             6   *! *!
Other  *!  *!             *!  *! *!
```

Note that the presence of '!' after all non-value rules will collapse all flows on to one of the following flow tuples:

```
0  224.0.0.0  17  0  0  MC-UDP
0  0          17  0  0  remaining UDP (UC)
0  0          6   0  0  TCP
0  0          0   0  0  Other
```

7.2.3 Fs2flows

Note that fs2flows is a simple extension of FS2flows-1.0 which has the following identification:

Hans-Werner Braun <hwb@nlanr.net> flow analysis concepts and initial code, in conjunction with kc Claffy (kc@nlanr.net). Major rewrite by Kevin Fall <kfall@sdsc.edu>, with further modifications by Joel Apisdorf <apisdorf@mci.net>.

In particular `dostats.c` `dostats.h` `queue.c` `queue.h` `trace.h` `tvops.c` `tvops.h` `types.h` `util.c` `util.h` are essentially unchanged, while `fs2flows` has been extended to do rule matching and summaries and the change the output format for my own use.

This program converts an `fs` format file (set of logged IP packets, e.g. from `tcpdump2fs`) to a "flows" file, with information on flows of packets, rather than individual packets. Using this program is the first step in any flow-based analysis. At this stage you also determine the finest temporal granularity which you will be able to use (i.e. time interval for a time-series information).

Usage:

`fs2flows` [options]

-i <int> : period in seconds for partial flow information = temporal resolution of output flow information. Also period for internal housekeeping.

-r <rulefilename> : optional rule set to use internally before generating output. Not normally used - only if the output would be obscenely large otherwise. Normally better to subsequently apply `flowfilter`.

-f <filename> : file to read `fs` record from, else standard input.

-o <string> : objname, i.e. kind of flows to extract. One of:

sh - source host

sn - source network (A/B/C class based)

dh - destination host

dn - destination network (A/B/C class based)

np - network pairs (A/B/C class based)

hp - host pairs

pq - port quintes (saddr, daddr, prot, sport, dport)

We almost always make use of `pq`; the others can then be derived more flexibly using `flowfilter`.

-p : print useful output (!) (normally used)

-k : fillin not wanted - avoid if you want temporal records.

-t <int> : flow timeout in seconds, i.e. number of seconds without a packet after flow is assumed to have ended (important).

-1 <hexlongword> : src address

-2 <hexlongword> : dst address

-3 <int> : ip protocol number

-4 <int> : src port

-5 <int> : dst port

For example:

```
cat FILE.fs | fs2flows -o pq -p -i 10 -t 100 >! FILE.flows
```

Output all flows based on port quintes, assume a flow has ended when no packets are seen for 100 second, output flow time series information every 10 seconds.

7.2.4 Flowfilter

`Flowfilter` is applied to flow files in order to filter and/or summarise the flows which they describe. This is the most important analysis tool – the virtual traffic scalpel.

Usage:

-a <float> : minimum time (unix seconds) - will include flow/records which overlap this time.

-b <float> : maximum time (unix seconds) - will include flow/records which overlap this time.

-f <filename> : output only flows which match rule(s) in the specified rule file. Do NOT change flows as a result of '!' rules.

-r <filename> : summarise flows using rules in the specified rule file. Non-matching flows are retained.

Reads from standard in and writes new flow file to standard out.

Note: multiple -f and -r options can be specified, in which case they are applied to each flow in the order in which they were specified.

E.g. if the multicast/unicast rule file in the section on rules, above (section 8.2.2), were called "mc-uc.rules" then the following would create a summary set of flows which only distinguished between UDP multicast, UDP unicast, TCP and all other traffic.

```
cat FILE.flows | flowfilter -r mc-uc.rules >! FILE.flows.mc-uc
```

7.2.5 flow2data

This program processes a flow file which include 'I' and 'T' records (i.e. partial flow records, generated by the "-i" option of fs2flows) in order to generate data file with bandwidth against time entries for use with gnuplot, etc.

Usage:

```
flow2data flowfile [period-count [start end]]
```

Period-count (default 1) specifies the number of 'I' record periods which contribute to each time period in the output file. E.g. a flow file with 1 second granularity (made using "fs2flows -i 1") can be converted to data for average bandwidth in minute samples using a value of 60. N.B. this is a multiple of the value used for fs2flows, NOT an actual time in seconds.

Start and end times are in unix seconds, as usual.

Notes:

- the output file has time in the first column. Time zero is the first output time, typically the first time in the flow file.
- the output columns are determined by the rule numbers used to generate the flows. Specifically, a flow generated/matched by rule 1 will be in column 3, rule 2 in column 4, etc. Any flows unmatched by any rule will be recorded in column 2. Rule numbers are the position within the rule file of the corresponding rule (see also flow file format notes, above).

7.2.6 Summary

Summary is a perl script which prints a pretty table (in descending byte size order) of flows in a given flow file.

Usage:

```
summary <flowfile> [<starttime> <endtime>]
summary-short <flowfile> [<starttime> <endtime>]
```

If no start/end time is specified then the 'A' records in the flow file are used to generate information on whole flows. Otherwise 'T' records are used, and start/end time accuracy is limited to individual T record periods.

The output format for summary is:

```
Type      #flows      bytes      %      packets      % dur(s)  rate bit/s  pkt-size
-----
```

The output format for summary-short is:

```
Type      bytes      % rate bit/s
-----
```

Notes:

The type field is taken from the rule name.

When multiple flows have the same rule name/type they will be included individually in the table with an additional explanation line giving detailed information on the flow. This is normally a diagnostic check or used as part of the (often iterative) process of refining a rule file.

Note that bit rates are for the measured duration of the flow, rather than the file as a whole. Consequently, bit rates will be reported differently when using A and T records, and for different levels of summarisation. Also, the bit rates will not normally add up to the 'total' bit rate (which typically has a different overall duration).

This is derived from the similar script included with FS2flows-1.0.

7.3 User analysis tools

These tools are intended for use in analysing user activity, e.g. amount and correlation of movement, etc. These are derived from original MASSIVE1/ITW analysis and tools. After describing the common file format, the following tools are described: `sort_by_time`, `process_simple`, and `process_correlation`.

7.3.1 File format

`process_simple` and `process_correlation` read a file made up of lines of the form:

```
unix-time ip-address keyword qualifier
```

where:

- `unix-time` is a float, in seconds;
- `ip-address` is just a unique string to identify a user;
- `keyword` is one of:
 - "start-moving" - user begins to move
 - "stop-moving" - user stops moving
 - "start-stationary" - user is present and stationary
 - "stop-stationary" - user is no longer stationary (moving or gone)
 - "start-speaking" - user begins to speak (limited use by `process_simple`, only)
 - "stop-speaking" - user stops speaking (limited use, by `process_simple`, only)
 - "enter" - user arrives in named world
 - "leave" - user leave named world
- `qualifier` is ignored for all but `enter` and `leave`, when it is a string world name which is being entered or left.

NOTE:

For most tools, audio is described as "start-moving" "stop-moving" "enter" and "leave", respectively, in a disjoint data file.

7.3.2 Generating files

There are a number of tools for converting log files from different systems and applications into the common form used by the tools in the following section to analyse user activity.

For example, the OOTW world manager process can optionally log information about used in the world, such as movement, appearance of graphical mouth. The tool `ootw2stats` converts this into the common stats file format for subsequent analysis.

```
Usage: cat <ootw-file> | ootw2stats >! <move-file>
```

7.3.3 Sort_by_time

This program sorts a file in the above format by timestamp (using `qsort`) with the additional constraint that events at the same timestamp preserve a sensible order with respect to `enter/leave/start/stop`. This is because `stop` must precede `leave` and `start` must follow `enter`.

This is mainly used after concatenating multiple log files from different users and/or sessions to perform a global analysis.

7.3.4 Process_simple

usage:

```
process_simple [options] <infile>
```

-o <string> : operation to perform on file, one of:

"move" : % time moving, % vs incidence (person-seconds)

"stationary" : length (seconds) of individual stationary period, distribution, time vs incidence (person-seconds)

"worldreturn" : length of time (seconds) out of world before returning, time vs incidence (person-seconds)

-P <filename> : participant id filename, implies participant number.

-h <float> : movement "hold" time in seconds, i.e. merging of small stationary periods.

-W <string> : world name, if specified, restricts "move" output to activity in named world only. N.B. does not affect other operations.

-N <int> : participant number (participant id filename specified) or world number (no participant id filename specified).

- For participant number, specifies (for "move" operation, only) number of participants in participant file to record information for (required with participant file).
- For world number, specifies (for "move" operation only) the number of world (in order of discovery) to record information for.

-p <string> : participant. For "move", produces output only for named participant (ip-address).

-b <int> : initial number of buckets, automatically increased (default 100), mainly for "move", which distributes % moving between specified number of buckets.

-c : cumulative flag - output data is cumulative rather than simple bucket distribution.

-t <float> : bucket time (default 5.0), not used with process_simple.

-s <float> : bucket size (default 10), time period in seconds associated with a single bucket (i.e. temporal resolution) for "stationary" and "world return".

<infile> : of above format.

Optional "participant_file" (-P <file>) is list, one per line, of ip-address values. Line number is then numeric id for that user.

7.3.5 Process_correlation

Usage:

```
process_correlation [options] <input-file>
```

Options:

-o <string> : operation, one of:

- "worldi" - world interarrival time, seconds vs incidence, distribution (at granularity of bucket_size seconds) of interarrival time for consecutive entry events for individual worlds.
- "worlda" - number of entry events in each bucket_time seconds interval, from the start of the file.
- "worldp" - world population, people vs incidence (s), i.e. number of people in a single world at the same time, for all concurrent worlds.
- "people" - people present in the system (any world) at the same time vs. incidence (s - system up time)

Multimedia Networking for Inhabited Television - University of Nottingham - 1999

- "synthp" - synthesised random world populations, people vs incidence (s). World populations (c.f. "worldp") assuming random distribution between all world encountered.
 - "move" - number of people moving simultaneously [in buckets] vs incidence (s)
 - "synthm" - synthesised random number of people moving simultaneously vs incidence (s), c.f. move but assuming same profile of time vs. number present and single overall average %age time moving for everyone.
 - "groupsize" - size of groups entering worlds, people vs incidence (occasions)
 - "groupsize2" - size of groups entering worlds, people vs incidence (person-occasions)
 - "groupspread" - delay of group member entering world, seconds vs incidence, at granularity of bucket_size seconds.
 - "plotworld" - generate eps visualisation of movement and world transitions. Requires local files "plotworld.eps.header" and "plotworld.eps.trailer" to build output (taken from gnuplot epsfiles).
- b <int> : number of buckets (default 100), would be correctly autosized from 0 for all operations except synthp and synthm, which might need a larger than default number if actual people per world significantly anticorrelated.
- c : cumulative flag, output cumulative values rather than point distribution.
- t <float> : bucket time (default 5.0), used in "worlda" operation only.
- s <float> : bucket size (default 10), used in "worldi" and "groupspread" to determine temporal granularity.
- T <string> : title for plotworld visualisation.
- S <float> : min. time offset, relative to first entry in file, - starting point of plotworld visualisation.
- E <float> : max. time offset, relative to first entry in file – end point of plotworld visualisation.
- <input-file> : in above format.

8 References

- [1] <http://www.iet.unipi.it/~luigi/research.html>
- [2] Luigi Rizzo, “*An Embedded Network Simulator to Support Network Protocols’ Development*”, 9th International Conference on Computer Performance Evaluation: Modelling Techniques and Tools, St. Malo, France, June 1997, LNCS-1245, pp. 97-107, Springer Verlag.
- [3] <http://ee.lbl.gov/>
- [4] <http://moat.nlanr.net/>
- [5] Kimberly C. Claffy, Hans-Werner Braun and George C. Polyzos, “*A parameterizable methodology for Internet traffic flow profiling*”.
- [6] <http://www.auckland.ac.nz/net/Accounting/ntm.Release.note.html>
- [7] <http://www.ietf.org/html.charters/rfm-charter.html>