On Consistency and Network Latency in Distributed Interactive Applications: A Survey—Part II

Abstract

This paper is the second part of a two part paper that documents a detailed survey of the research carried out on consistency and latency in distributed interactive applications (DIAs) in recent decades. Part I reviewed the terminology associated with DIAs and offered definitions for consistency and latency. A classification for consistency maintenance mechanisms was given and various mechanisms belonging to the first of three categories, time management, were described. Here, in the second part of the paper, the remaining two categories of mechanisms are examined—information management (such as predictive contract techniques, relevance filtering, packet bundling) and system architecture (such as QoS and protocols).

1 Introduction

Despite increased processing power at participating nodes and the availability of greater communications bandwidth, the fundamental limitation to the deployment of distributed interactive applications (DIAs) is the problem of maintaining a consistent worldview in the presence of latency and jitter. In Part I of this paper, the link between consistency and a number of aspects of DIAs such as temporal and spatial synchronization, causality or ordering, and concurrency was explored. The link with responsiveness and fidelity was also explored, in that poor consistency can lead to problems of fidelity, and fidelity is often sacrificed to maintain both consistency and responsiveness. To avoid inconsistencies such as divergence, causality violation, and expectation violation, various techniques and methods have been proposed and implemented over the years. We refer to any element employed to ensure a sufficient, uniform dynamic shared state for all participants in a DIA as a consistency maintenance mechanism.

Three categories of these mechanisms were identified and the first of these, time management techniques, was explored in Part I of the paper (Delaney, Ward, & McLoone, 2006). Here, we explore the remaining two categories—information management and system architecture. It should be noted that the techniques are not mutually exclusive and the DIA designer can mix techniques across categories to suit each particular application. The paper ends with some concluding remarks.

2 Consistency Maintenance Mechanisms

2.1 Information Management Techniques

2.1.1 Predictive Contract Agreement Mechanisms: Dead Reckoning. Predictive contract agreement mechanisms are optimistic consistency maintenance mechanisms that operate by employing a form of controlled inconsistency (Zeigler, Cho, Lee, Cho, & Sarjoughian, 1999). All participants agree on a prediction algorithm, an associated threshold error, and a convergence algorithm. The threshold error reflects the amount of inconsistency that will be permitted to occur between the true state and the predicted state. When this error is exceeded, the inconsistent state is corrected by transmitting current state information to all other participants who must subsequently converge to this updated state. Predictive contract agreement mecha-

Declan Delaney*
Tomás Ward
Seamus McLoone
Department of Electronic Engineering
National University of Ireland, Maynooth
Maynooth Ireland
*Correspondence to diag_info@eeng.nuim.ie.
nisms reduce the amount of network traffic and therefore reduce the network latency.

In DIAs where much of the state information relates to the movement of entities, a polynomial predictor algorithm can be employed to reduce the amount of packets transmitted. The distributed interactive simulation (DIS) standard defines a dead reckoning protocol to achieve this. This protocol consists of a polynomial prediction algorithm and a convergence algorithm (IEEE, 1993; IEEE, 1995). Dead reckoning is a bandwidth saving mechanism and compensates for variable communication latency (Lin & Schab, 1994; Capin, Emeraldo, & Thalmann, 1999; Baughman & Levine, 2001). A synchronization message must be transmitted to adjust for inconsistencies that can result if dead reckoning is employed in the long term (Lui, 2001).

Dead reckoning has been described, analyzed, and modeled by a number of authors. It has been modeled as a Markov chain by Kwang-Hyun and Jong-Sung (2001) and as hybrid automata in (Ozutam & Oguztuzun, 1999), where it is called the players-and-ghosts paradigm (Blau, Hughes, Moshell, & Lisle, 1992). Durbach and Fourneau (1998) studied the influence of dead reckoning on network performance and on response time. They constructed a model of the interarrival times of packets on a real network and then represented the network as a Markov chain. They claim that their model of network packets is more accurate than those based on Poisson models.

The concept of dead reckoning has been extended to entity attribute extrapolation within the high level architecture (HLA) framework (Lin, Blair, & Woodyard, 1997; IEEE, 2000). It has also been enhanced by combining it with a priority round robin scheduling mechanism (Faisstnauer, Schmalstieg, & Purgathofer, 2000) and by employing a Kalman filtering approach to complex entities such as the human body (Capin, Pandzic, Thalmann, Magnenat, & Thalmann, 1997). Dead reckoning has been used to predict the occurrence of deterministic events. These events can then be transmitted to other users in advance and thus improve the consistency (Roberts, Strassner, Worthington, & Sharkey, 1999; Krumm-Heller & Taylor, 2000).

Variations on the standard DIS dead reckoning algorithms have been proposed. Singhal and Cheriton (1995) described a position history-based dead reckoning protocol. This is a hybrid technique that employs a first order polynomial estimator for very low and very high acceleration. Otherwise second order parabolic estimation, which uses the three most recent packet updates, is employed. Adaptive dead reckoning algorithms, based on variable threshold or variable prediction algorithm, have been described by various authors (Singhal, 1996; Bahadir & Halit, 1999; Cai, Lee, & Chen, 1999; Lee, Cai, Turner, & Chen, 2000). The dead reckoning protocol has also been customized to account for entity-specific characteristics (Bassiouni, Chiu, Loper, Garnsey, & Williams, 1998).

Duncan and Graćanin (2003) proposed a pre-reckoning algorithm which complements the dead reckoning approach. If a sudden change in movement is detected, such as a 90° turn, then an update packet is generated immediately instead of waiting for the dead reckoning prediction threshold to be exceeded. Group dead reckoning (Das, Singh, Mitchell, Kumar, & McGee, 1997) aims to prevent continuous flow of positional messages as a user moves through the environment of the DIA. It employs an interval-based scheme, with entities sending positions to a central controller at discrete intervals only if their position has changed. This controller then forwards positional data to other users as needed.

Delaney et al. proposed a hybrid predictive contract technique that chooses either a short-term deterministic dead reckoning model or one of a family of long-term statistically-based models (Delaney, Ward, & McLoone, 2003b; Delaney, Ward, & McLoone, 2003a). This results in a more accurate representation of the entity’s movement and a consequent reduction in the number of packets that must be communicated to track that movement remotely.

2.1.2 Relevance Filtering. Relevance filtering is based on the idea that most users of the DIA are only interested in a subset of the large volume of data available. It operates by analyzing the semantics of packets on transmission or reception and selecting only those which match a certain criterion (Bassiouni, Williams, & Loper, 1991; Bassiouni, Chiu, Loper, Garnsey, & Wil-
liams, 1997). It is often used in conjunction with multi-
cast protocols, so that a single source node simulta-
neously transmits to multiple destination nodes.
Relevance filtering is also referred to as interest manage-
ment, data distribution management, data flow manage-
ment, data filtering, and data subscription (Singhal,
1996; Morse, Bic, & Dillencourt, 2000).

The objective of relevance filtering is to control the
flow of data. Probably the most general model of data
flow is the aura-nimbus model proposed by Benford and
Fahlén (1992; Benford & Fahlén, 1993). The essential
components of the model are the medium, the aura, the
focus and the nimbus (Greenhalgh & Benford, 1997).
The medium is a communication type such as audio,
visual, or text. The aura is an object- and medium-
specific subspace in which interaction (communication)
may occur. For example communication might be re-
stricted to audio only; hence an audio aura. Entities
must have intersecting auras to communicate. Within
the aura different levels of awareness can be established.
To achieve this each entity is given a focus and a nim-
bus. The focus defines the sphere of influence of the
entity; the nimbus defines the sphere of interest. For
example, if the nimbus of entity 1 intersects the focus of
entity 2, then entity 1 has awareness of entity 2 and en-
tity 2 can thus share state information with it. The nim-
bus and focus spheres can be defined by characteristics
such as sensor capability, entity type, spatial distance,
location and social awareness. The spatial model was
extended by the introduction of third party objects,
which are independent objects in a virtual world that
perform various awareness adaptations and aggrega-
tions (Benford, Greenhalgh, & Lloyd, 1997; Benford &
Greenhalgh, 1997). These objects can adapt the level of
awareness between DIA users as a function of the task
being performed and they allow the application of filters
to reduce level-of-detail or the aggregation of data. Ag-
gregation (e.g., creating a new group object from indi-
vidual objects) is also referred to as granularity (Roberts,
2004). The objective of third party objects is to facili-
tate construction of the DIA and improve scalability in
interactive environments, while maintaining consistency.
Most relevance filtering techniques can be related to this
model, with refinements to deal with the scalability is-
sue. However, if this model is implemented as de-
scribed, it doesn’t scale to large numbers of active enti-
ties (Benford & Greenhalgh, 1997), as all entities need
to be examined for intersecting aura, nimbi, and foci
and thus the consistency of the DIA is gradually lost.

In developing a taxonomy of relevance filtering tech-
niques, Morse et al. (2000) identified two overall ele-
ments in every relevance filtering technique:

1. An implementation architecture for performing
   the filtering;

To express interest, relevance filtering employs a con-
cept known as interest expressions (Morse et al., 2000).
These define the data the entity is interested in receiving
using a specific syntax. Processes accept subscriptions
from user entities, evaluate interest expressions and use
them to filter messages in accordance with user needs
expressed in the user subscription. Dedicated processes
called interest managers may be used for achieving this.
The filtering process may be viewed as being either ex-
trinsic or intrinsic. When the data is filtered based on
examination of the header properties of the transmitted
packets, such as the packet’s source/destination ad-
dresses, it is referred to as extrinsic filtering. When the
filtering is performed by analyzing the application-
specific contents of the packet it is called intrinsic filter-
ing. Extrinsic filtering is quicker to perform but less pre-
cise than intrinsic filtering.

Various selection criteria have been employed in in-
terest expressions to define the data of interest:

1. Occlusion or culling, which might exclude all in-
formation out of visual or audio range (Funk-
house, 1995; Roehle, 1997);
2. some physical attribute such as spatial proximity,
location, social awareness, sensor type, tempera-
ture or radio transmission frequency, or entity type
(Bassiouni et al., 1997; Morse et al., 2000).

There are various filtering implementation architec-
tures. In the RING system, all entity updates are sent to
a central server and the server only forwards updates to
those nodes with entities that are visible to each other
(Funkhouser, 1995). Similar to this approach is area of
interest (AOI) data management (Macedonia, Zyda, Pratt, Barham, & Zeswitz, 1994). In MASSIVE the AOI corresponds to the aura (Greenhalgh & Benford, 1995). AOI data management is an intrinsic filtering scheme and it requires all nodes to relay entity state data to one or more central managers called area of interest managers (AOIM) or subscription managers. Nodes must also inform the managers of the data they are interested in receiving. The managers then decide what data to forward to each node based on some selection criteria such as distance (Bassiouni et al., 1997; Anthes, Heinzlreiter, & Volkert, 2004). Unicast communication is used because AOI produces a customized stream of data for each node.

Another implementation architecture employs multicast communication. In this case data is transmitted to multicast groups and nodes must join or leave a group to start or stop receiving data from that multicast group. It is important to partition the data efficiently among the multicast groups to avoid nodes having to join several different groups, each containing a small quantity of essential data. In addition the number of join/leave operations must be minimized as these increase network traffic, consume computing resources and hence cause latency to increase. Data may be partitioned so that it is associated with an area of the virtual environment or with an entity (Barrus, Waters, & Anderson, 1996b; Rak & Van Hook, 1996; Zou, Anmar, & Diot, 2001). In entity-based filtering, each entity is allocated a different multicast network address. An entity multicasts its state information to another entity if their auras intersect and if it has previously subscribed to receive information from it. An example of this kind of filtering is the work by Lee, Yang et al. (2000). Grid-based filtering divides the DIA into cells and each cell is assigned a multicast network address. An entity multicasts its state information to another entity if their auras intersect and if it has previously subscribed to receive information from it. An example of this kind of filtering is the work by Lee, Yang et al. (2000). Grid-based filtering divides the DIA into cells and each cell is assigned a multicast network address (Macedonia, Zyda, Pratt, Brutzman, & Barham, 1995; Capps & Teller, 1997). A group of participants is associated with each grid cell. A participant joins the multicast group associated with each cell its AOI intersects (proximity detection) (Abrams, Watsen, & Zyda, 1998). Thus, multicast groups change as participants move around the DIA and both the filtering efficiency and management are dependent on the grid size (Morse et al., 2000). The shape of the regions or cells can be rectangular as in the ModSAF system (Russo, Shuette, Smith, & McGuire, 1995; Rak & Van Hook, 1996). To reduce the number of grid cell meeting points to three, the cells in NPS-NET are hexagonal in shape (Macedonia et al., 1995). In SPLINE the cells are called locales and they can be any shape or size, so that the cells can be built around the structure of the virtual environment (Barrus, Waters, & Anderson, 1996a; Barrus et al., 1996b). The grid-based and entity-based filtering approach can be combined to achieve a finer partitioning of the data as in the three-tiered management employed in Abrams et al. (1998).

The use of static cell sizes will break down if all entities are located in a sufficiently small area of the DIA. One approach to solving this is to dynamically modify the size of the regions as in Abrams et al. (1998). Each region receives low-fidelity information about entities such as position, speed and orientation. This information can then be used to gather more high-fidelity information. Another approach to solving this problem is to dynamically adapt the AOI as the density of objects in the AOI either increases or decreases (de Oliveira & Georganas, 2003). Robinson et al. (2001) shift the boundaries between the areas of interest so that users are balanced among areas. Han, Lia, and Lee (2000) proposed a scalable interest management technique based on spatial distance and user interests. Their technique also deals with crowd clustering in a specific area of the DIA environment and shows a significant improvement in scalability performance of large-scale DIAs compared to existing approaches.

Another scalable interest management technique is used in MASSIVE-2 (Benford & Greenhalgh, 1997b; Benford, Reynard, Greenhalgh, Snowdon, & Bullock, 2000). Here third party objects are employed to construct the spatial structure of the virtual world. Hence there is a dynamic hierarchy of group objects that maps directly onto the virtual world. Each of these groups has a multicast address and entities join/leave different groups as they move about the virtual environment. In addition the awareness level can be adjusted to match the task being performed by each group.

Relevance filtering software is normally deployed on a
central server that relays relevant data to participating nodes based on a set of criteria and knowledge of the state of entities at each node in the DIA. However, if a central server is not employed, there is the paradox that in order for entities to know when not to transmit updates to each other, they must continuously update each other on their current state (Makbily, Gotsman, & Bar-Yehuda, 1999). To reduce the number of update packets in this case, Makbily et al. proposed a filtering approach based on update-free regions for pairs of entities. These regions are mutually irrelevant regions in user-parameter space, when interaction between the two users is not possible. This idea exploits the visual relevance relationship (based on the criteria of proximity, visibility, and direction) between users. It builds on the concept of a potentially visible set (PVS) (Airey, Rohlf, & Brooks, 1990). A similar, though more robust implementation called frontier sets was proposed by Steed and Angus (2004; Steed & Angus, 2005). The objective in both these proposals is to increase the scalability of the DIA by reducing the quantity of update packets; hence network latency is reduced and consistency can be maintained despite additional data throughput. Bassiouni et al. (1998) also examined the issue of decentralized filtering in the form of gateway severs, a kind of hybrid centralized/peer-to-peer system, referring to the resulting inconsistencies as filtering errors. They proposed, designed, and evaluated a number of reliable filtering methods such as gateway dead reckoning, periodic broadcast, and reachability range. They found that each of the methods they proposed resulted in additional network traffic, with a trade-off between network traffic and global consistency of the DIA.

One problem that limits the implementation of relevance filtering is the limited number of IP multicast addresses available in internet protocol version 4 (IPv4) (de Oliveira & Georganas, 2003). This problem should be solved when internet protocol version 6 (IPv6) becomes widely deployed, whenever that may be. Morse et al. (2000) noted the restrictions imposed by multicast hardware and the overhead involved in configuring and reconfiguring groups and suggest that in some cases this additional processing may add to the latency problem.

Cai et al. (1999) employed relevance filtering concepts to determine error thresholds for dead reckoning algorithms. They established four threshold levels based on an entity’s AOI and its sensitivity region. When the sensitivity region of one entity intersects the AOI of another entity a collision is almost certain to occur. Their experiments showed a reduction in transmitted packets without a reduction in extrapolation accuracy.

The performance and reliability of relevance filtering has been comprehensively documented by Bassiouni et al. (1997). They evaluated different design alternatives of filtering at transmission and filtering at reception for a spatial distance filtering criteria. Filtering at transmission provides the greatest bandwidth saving, whereas filtering at reception only serves to filter incoming messages at a gateway node for a local network. They demonstrated a significant reduction in transmitted packets for various AOI distance values. The emphasis of their work was very much on bandwidth usage. They did not address the issue of latency. The issue of consistency was addressed by examining the filtering errors introduced when data that should not have been filtered is filtered. To combat this they make a number of proposals for reliable filtering schemes. Interested readers are referred to the paper for more details (Bassiouni et al., 1997).

2.1.3 Data Compression. Although data compression has the same goal as relevance filtering, they are quite distinct mechanisms (Singhal, 1996). Data compression is a technique for the efficient storage and transmission of data (Bassiouni, 1983; Bassiouni and Mukherjee, 1990). In DIAs, the objective of compression is to reduce the size of the transmitted packets. Packets should be compressed at the transmitting node and then decompressed at the receiving node.

There are two main categories of compression techniques (Singhal & Zyda, 1999):

1. Lossless compression: reduces the amount of data by changing the encoding format for the packet while ensuring that there is no loss in information;
2. Lossy compression: may eliminate some of the redundant or irrelevant data from the packet. Dead reckoning may be viewed as a lossy compression as
only a sampled version of the entity’s movement is transmitted to other nodes.

Each of these types of compression may involve internal or external compression. **External compression** operates on the data in the current packet and previously transmitted packets. **Internal compression** operates on the data in the current packet only. For internal compression to be useful the following inequality must be true:

\[
\frac{P_L - P'_L}{B} \geq 2T_c
\]  

where
- \(P_L\) is the original message size;
- \(P'_L\) is the message size after compression;
- \(B\) is the bandwidth;
- \(2T_c\) is the time taken to execute the compression/decompression algorithms.

The type of compression employed is generally dependent on the data being compressed and hence on the application domain. However, PICA (protocol independent compression algorithm) is an application-independent external compression technique that reduces the bit rate in DIAs (Bassiouni et al., 1997). It operates by sending only the byte difference between the current state of an entity and a reference state.

Interestingly, most modern modems compress outgoing data to optimize bandwidth usage, but this tends to increase latency as the modem needs a minimum amount of data before it can compress and send data. In other words, the inequality in Eq. 1 is violated. There is a hardware timeout if no further data is forthcoming (Cheshire, 1996). Manufacturers such as 3Com and Motorola have developed modems for entertainment DIAs that send data packets instantaneously, thus avoiding the latency associated with data buffering at the network layer and illustrating the importance of application layer compression for DIAs.

### 2.1.4 Packet Bundling

Packet bundling or **aggregation** is another data management technique, motivated by the fact that networks can only handle a limited number of packets per unit time (Hook, Newton, & Fusco, 1994). The idea is to reduce the number of packets transmitted, in contrast to packet compression, which seeks to reduce the size of each transmitted packet. It involves assembling a number of individual packets into a larger data unit and transmitting this new unit as a single packet. Most routers, bridges, and gateways can only process a fixed number of packets per unit time. Thus, by grouping a number of packets together, the network can handle more data per unit time. There is also a reduction in the amount of data transmitted, as a single packet header/trailer replaces the headers/trailers of a number of individual packets.

The size of the bundled packet is determined by the network, and is usually limited to the maximum Ethernet packet size of 1500 bytes. Thus, to transmit a large packet the network uses a process known as fragmentation. Fragmentation introduces the additional issues of fragmentation strategy and reconstruction strategy, introducing additional processing overhead (Tanenbaum, 1996). Bundling also introduces latency because the transmission time of some packets will be delayed during the bundling process so that other packets can be included. This may make it more difficult to maintain DIA state consistency, although packets containing time critical information do not have to be bundled. There is also a processing overhead involved in assembling and segmenting the packets at both the source and the destination. As a result there is a trade-off between bandwidth, packet size, and latency. This trade-off can be managed by using a timeout policy (which means that there is an upper delay limit), a quorum-based policy (a minimum number of packets to bundle) or a combination of both (Singhal & Zyda, 1999).

As part of their overview of the main techniques used to reduce bandwidth usage in entity-based virtual simulations, Bassiouni et al. (1997) investigated packet bundling in ATM-based DIS environments. They concluded that bundling under ATM is not rewarding because of the additional delay introduced by the segmentation (i.e. fragmentation) and assembly of packets. ATM cells are 48 bytes in size. They also found that the rate of increase in ATM delay with bundled packet size is greater than the rate of increase in time needed to transmit the bundle.
Liang, Cai, Lee, and Turner (1999) proposed and studied the performance of two packet bundling techniques employing single and multiple priority queues. Their approach defines four levels of priority according to the nature of the data—time critical data is given high priority, heartbeat data (periodic messages sent by entities to indicate they still exist) is given low priority. Each priority also has an associated timeout; high priority data has a low timeout. With multiple priority queue, bundles with low priority packets are allowed to be larger than bundles with high priority packets. Bundles with high priority packets will always be transmitted first. In the single priority queue, all packets are queued in order of priority. Bundles are generated by taking packets from the queue until either the maximum packet size is reached or a timeout occurs for one of the packets in the bundle. In this case a balance must be reached between bandwidth, bundle size, and timely delivery. Their simulation results show a reduction in the average packet transmission rate and average bandwidth requirement, but an increase in the overall latency time.

2.2 System Architecture Techniques

DIAs require sufficient consistency to make interaction appear realistic across large distances. However, network latency makes it difficult to satisfy this requirement. Techniques for reducing the quantity of information that must be communicated across the network aim to reduce the impact of latency. Time management techniques seek to synchronize data so that users share the same view of the application state. For collaborative interaction, this synchronization is time critical and so the effects of network latency must be overcome. The implementation of these techniques exploits a number of concepts, technologies, and approaches that aim to improve the efficiency of processing and disseminating data among participating nodes of the DIA and that therefore impact the consistency of the application and the effects of network latency.

2.2.1 Network Architecture. A DIA consists of a number of computers that are interconnected and may be geographically dispersed. How these computers are connected affects the management of data and hence the ability to maintain consistency and reduce the effects of network latency. There are two basic architectural topologies: client-server (c-s) and peer-to-peer (p2p).

In p2p architectures information is transmitted directly between participants (Frécon & Stenius, 1998; Balikhina, Ball, & Duce, 2002; McGregor, Kapolka, Zyda, & Brutzman, 2003). Each participant maintains a replicated version of the distributed world and consequently consistency is difficult to maintain (Dourish, 1995). However, this architecture facilitates local responsiveness and minimizes consistency problems associated with concurrency. Examples of p2p architectures include DIS (IEEE 1993), MASSIVE (Greenhalgh & Beaford, 1995; Greenhalgh, 1999), DIVE (Carlsson & Hagsand, 1993), Mr. Toolkit (Shaw, Green, Liang, & Sun, 1993), and NPSNET (Macedonia et al., 1994).

In contrast a c-s architecture provides a central authority to which all participants connect. The server maintains the definitive state of all entities and controls distribution of information between participants. The client nodes are responsible for rendering information to the participant and updating the server with changes. This makes it easier to maintain consistency, but with reduced responsiveness due to network latency. Consistency becomes more difficult to maintain as the application scales up and the quantity of data to be managed increases. This is because the server acts as a bottleneck. In addition the operation of the entire DIA hinges on a single server. Examples of c-s architectures include RING (Funkhouser, 1995), VLNET (Pandzic, Lee, Thalmann, Capin, & Thalmann, 1997), and NetEffect (Das et al., 1997). In BrickNet the virtual world database is split over a number of nodes, with a server acting as broker to keep track of what part of the world is being maintained by each node and relaying sufficient information to maintain the level of consistency requested by each node (Singh, Serra, Png, & Ng, 1994).

The p2p and c-s architectures may be combined to produce hybrid architectures. Examples include SPLINE (Waters, Anderson, Barrus et al., 1997), Community Place (Lea, Honda, Matsuda, & Matsuda, 1997) and
PaRADE (Roberts, Sharkey, & Sandoz, 1995). Multiple servers with coordinated data flow between them can provide scalability and redundancy. The servers themselves are connected using a p2p architecture and participants connect to only one server. The servers then coordinate their activities to relay information between interested participants. Multiple server system architectures introduce additional latency to the system as well as having to perform more processing of the data received from participants and other servers (Singhal & Zyda, 1999). A hybrid network topology to facilitate tightly coupled collaboration was proposed by Anthes et al. (2004). They defined an entity hierarchy consisting of three levels—clients, domains, and clusters. Each level in the hierarchy is responsible for certain tasks. Domains can be split or merged as the number of clients increases or decreases. To facilitate closely-coupled collaboration temporary p2p connections are established between clients by their domain server; prediction based on a spherical AOI is used to decide the necessity for a p2p connection. State updates use a hierarchical message distribution scheme, beginning with clients, then domains, and finally clusters, which then multicast the message to other domain servers.

2.2.2 Communication Protocols. DIAs rely on the transfer of data between participating nodes in the shortest time possible to ensure a sufficiently consistent worldview for all participants. Regardless of the network architecture, this data must be communicated unambiguously between nodes. A protocol is an agreement between the communicating parties (nodes) on how communication is to proceed (Tanenbaum, 1996). This agreement can be taken at different layers in the ISO OSI reference model. Normally, protocols for DIAs are defined at either the application layer or the transport layer. The most commonly supported network layer protocol is IP (internet protocol).

The most common transport layer protocols are the transport control protocol (TCP), which guarantees reliable transmission across a distributed environment, and the user datagram protocol (UDP), which operates on a best-effort basis (Tanenbaum, 1996; Choukair & Retailleau, 2000). The impact of network latency on the DIA when employing TCP and UDP depends on the type of interaction occurring in the DIA. For short-term interactions, when the amount of information is small and it needs to be transmitted on an irregular basis, UDP is preferred over TCP; for long-term interactions, when data needs to be sent regularly or a large amount of data needs to be transmitted, then TCP is to be preferred. There is thus a trade-off between the quantity of data to be transmitted and the regularity of transmission. This is a consequence of the overhead involved in setting up and destroying network connections.

To overcome the unreliability of UDP, DIS (IEEE, 1993) and SPLINE (Barrus et al., 1996a) send messages with complete object state. In this way even if some messages are lost, consistency can be restored when the next message arrives. A new protocol, called the stream control transmission protocol (SCTP), is similar to TCP as it provides reliable full-duplex connection (Caro et al., 2003). However in addition it offers new delivery options that suit multimedia applications such as multistreaming, allowing independent delivery among data streams, multihoming, allowing end points of a single association to have multiple IP addresses, and partial reliability, allowing each message to be assigned a delivery reliability level.

TCP and UDP are examples of IP unicast protocols, sending packets to one destination node at a time. DIAs, however, may potentially have hundreds of thousands of simultaneous participating nodes. If each participant were to transmit information directly to every other participant, the interconnecting networks would quickly become congested and it would be impossible to maintain a consistent application with increased scalability for the sending node. To resolve this inefficient communication, IP multicast protocols are employed (Deering, 1989; Deering & Cheriton, 1990). IP multicast is a network layer protocol that provides simultaneous unreliable transmission to multiple destination nodes. In DIAs, multicast is used extensively in relevance filtering, with each group being assigned a unique multicast address. It is also used whenever messages need to be sent to multiple nodes efficiently to increase the scalability of the DIA (Macedonia et al., 1995). A number of disadvantages of IP multicast have been doc-
multicasting: it is difficult to implement efficiently on a point-to-point medium; many internet routers are not multicast aware or are limited in the number of groups they can support due to issues such as addressing, congestion control, and administration (Morse et al., 2000; Morse & Zyda, 2001; Fisher, 2002; Li & Zhang, 2002; Auerbach, Gopal, Kaplan, & Kutten, 2003; Hosseini & Georganas, 2004). These prompted the development of overlay multicast and reliable multicast protocols.

Overlay multicast was developed to provide IP multicast capability over networks that don’t offer multicast capability at the network layer and where it is usually implemented at the application layer. The most widespread overlay network is the multicast backbone or MBone (Eriksson, 1994). The DIVEBone extends the MBone and is an application-level network architecture built as a stand-alone part of the DIVE toolkit (Frécon & Stenis, 1998). A good overview of application layer multicast (ALM) is given in (Hosseini & Georganas, 2004).

Most authors acknowledge the fact that different multicast applications have different reliability requirements (Floyd, Jacobson, Liu, McCanne, & Zhang, 1997; Pullen, 1999; Singhal & Zyda, 1999; Li & Zhang, 2002; Auerbach et al., 2003). As a result, several reliable multicast approaches have been proposed. Reliable multicasting protocols refer to error-free eventual delivery of information to all the applications with some level of ordering (Li & Zhang, 2002). Floyd, Jacobson, McCanne, Liu, & Zhang (1995) and Floyd et al. (1997) describe a scalable reliable multicast protocol for a whiteboard application where each member of the multicast group is responsible for its own correct reception of all the data using a time-out mechanism. Kasera et al. (2000) also examined the scalability of reliable multicast transport but using active services. Pullen looked at the issue of reliable multicast in DIAs and detailed the workings of the selectively reliable transport protocol. This protocol employs three service modes according to the requirements of the DIA—best-effort multicast, reliable multicast, and reliable datagram. Sato, Minamihata, Fukuoka, and Mizuno (1999) also proposed a reliable multicast protocol suitable for DIAs that uses the concept of mutual aid regions (MAR).

Each MAR has a node that is responsible for acknowledging receipt of packets in that MAR and that sends timeout requests to the nearest MAR if a packet is not received. A good overview of multicast protocols is given in Lao, Cui, Gerla, and Maggiorini (2005) and Vogel, (2004).

Many other application layer protocols exist that exploit the transport and network layer services to optimize delivery of application-specific data. The real-time protocol (RTP) provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio streaming, video streaming, or simulation data, over multicast or unicast network services (Schulzrinne & Casner, 1993; Perkins & Crowcroft, 2000; Schulzrinne, Casner, Frederick, & Jacobson, 2003). RTP does not address resource reservation and does not guarantee quality-of-service for real-time services. An associated control protocol, RTCP, allows data delivery to be monitored and controlled. Mauve, Hilt, Kuhmunch, and Effelsberg (2001) adapted the RTP for interactive applications by inventing two new complementary protocols: the unreliable real time protocol for interactive media (RTP/I) and the real time control protocol for interactive media (RTCP/I). The RTP/I has four distinct data packet types that allow the communication of (1) events, (2) component states, (3) state changes, and (4) state queries.

Work performed by Zhang, Deering, Estrin, Shenker, and Zappala (2002) led to a signaling protocol designed to run over IP called Resource Reservation Protocol (RSVP). This protocol supports both unicast and multicast applications, allowing multiple senders to transmit to multiple groups of receivers and is the main protocol for integrated services or streaming multimedia (Tanenbaum, 1996). Kessler and Hodges (1996) proposed a protocol based on queue abstraction to communicate dynamic state information. They categorized communication between tasks of a virtual environment into three types and created an updateable queue for communication messages. Messages are prioritized and may skip or overwrite the queue if they contain a special key. Another protocol, the interactive sharing transfer protocol, was described by Waters, Anderson, Barrus, et al. (1997). This is a hybrid protocol supporting many modes of
transport for DIA data. It supports the sharing of information about objects in the DIA, streams audio data via RTB, and supports both unicast and multicast communication of state information between ISTP processes.

On the internet and world wide web, 3D virtual worlds can be constructed using the virtual reality modeling language (VRML). However, the underlying network support provided by the hypertext transfer protocol (http) is insufficient for large-scale DIAs. Brutzman, Zyda, Watsen, and Macedonia (1997) therefore proposed the virtual reality transfer protocol (VRTP) to facilitate peer-to-peer communications and network monitoring. Another application-independent network protocol, the distributed worlds transfer and communication protocol (DWTP), was proposed by Broll (1998).

Some standard protocols have been defined for DIAs. The DIS protocol (IEEE, 1993, 1995) strictly defines the information that can be carried by the protocol and limits its possible applications. The high level architecture (HLA) protocol tags data at the simulation design stage, not at run time, so that it combines dynamic definition of tags and flexibility in the definition of data content (IEEE, 2000). More recently, the introduction of XML (extensible markup language) by the W3C has led to the development of the simple object access protocol (SOAP) for communication between application programs (Jepsen, 2001). SOAP allows remote procedure calls (RPCs) between applications independently of the programming language and the operating system.

DIAs employ several different protocols. Modern DIAs employ a mixture of reliable and unreliable transport mechanisms, depending on how critical the data being transmitted is to the consistency of the DIA. At the application layer, protocols are application specific, although some efforts have been made to standardize these protocols. Protocols may also be downloaded and installed in real time by the application to ensure optimal communication between remote objects (Watsen & Zyda, 1998).

2.2.3 Quality of Service. The previous sections have described how the physical architecture of the underlying network together with appropriate protocols impact both latency and consistency. In this section the concept of quality of service (QoS) will be examined. QoS employs protocols to control the network technology so that a certain level of service is guaranteed for communications flow. A flow is a stream of data packets traveling between source and destination network layers (Tanenbaum, 1996) and it can be characterized by four primary parameters:

1. Network latency;
2. Latency variance or jitter;
3. Network bandwidth;
4. Transmission reliability.

These determine the QoS of the flow. Several techniques have been developed for achieving good QoS—over provisioning, buffering, traffic shaping, bucket algorithms, resource reservation, admission control, proportional routing, and packet scheduling. For streaming multimedia the IETF (Internet Engineering Task Force) has developed integrated services architecture (or flow-based QoS), employing the RSVP mentioned previously (Zhang et al., 2002). This protocol is used to communicate the requirements of the application to the network in a robust and efficient way. Another, simpler, architecture is the differentiated services architecture (or class-based QoS). Various schemes exist for managing the service classes such as expedited forwarding and assured forwarding. Another development is multi-protocol label switching (MPLS), a technique that constructs connection-oriented routes on the fly (Tanenbaum, 1996).

There is very little documented research relating DIAs and QoS. QoS may be seen as a set of constraints on the performance of the DIA such as those described by Houatra (2000); fault-tolerance, persistence, interactivity, and scalability. Houatra also describes software frameworks for the control and management of QoS in DIAs, by describing the Continuum project. Other attempts have also been made to define QoS architectures for certain aspects of the DIAs. For example Greenhalgh, Benford, and Reynard (1999) proposed a QoS architecture for managing streamed video within shared virtual worlds. In their proposal, QoS is driven by dynamically negotiating levels of mutual awareness among entities and by balancing group and individual needs.
They emphasize the dynamic nature of user demands in DIA and the consequent demands for dynamic QoS. How to configure the QoS is also an issue they raise—should a user be able to directly configure and reconfigure QoS for each entity in the DIA? Choukair et al. (2000) describe the implementation of QoS control in a DIA platform called VirtualArch which is built on the DVE collaboration model (DVECOM). The objectives of integrating QoS are to guarantee consistency and synchronization across the DIA. Choukair and Retailleau (2000) investigated the integration of QoS with real-time constraints (RT-QoS) into DVECOM.

 Provision of QoS over the internet is in a state of flux. For DIA, issues such as QoS setup times and reconfiguration times may increase network latency, reduce responsiveness, and hence make consistency more difficult to maintain. Other issues such as the need for QoS to be scalable and to adapt to real-time requirements, as well as being able to deal with the unpredictable nature of the data generated, must not be ignored.

2.2.4 Software Architecture. A DIA is ultimately a software program, albeit usually an extremely complex one. The design and deployment of the software is driven by the requirements of the DIA and the architecture of the underlying network (Singhal & Zyda, 1999). The quality of the software is therefore of fundamental importance and some of the key architectural issues that any successful DIA needs to consider are:

1. **Interactability**: How well can a user interact with the virtual world? This incorporates sensory input/output and real-time graphics rendering. Most consistency control mechanisms aim to meet this requirement.

2. **Interoperability**: How well can different DIA and associated technologies share information? There are two aspects of interoperability:
   a. The ability of distinct DIA and technologies to communicate. The HLA addresses this issue (IEEE, 2000);
   b. The ability of the same DIA running on heterogeneous operating systems to communicate.

3. **Dynamic extensibility**: How easily can the software system be updated and modified while executing? In the ATLAS-II application developed at CDS&N labs the objective is to allow the system extend itself automatically, rather than with explicit human intervention. In ATLAS this technique is referred to as **self-tunability** (Lee, Lim, & Han, 2002). When system degradation is detected or a system reconfiguration message written in XML is received, a component known as the resource discovery manager actively locates the new component to be loaded from either a local or remote source. The component is initiated and then registered with the appropriate ATLAS manager. In NPSNET V (Capps et al., 2000), all components are loaded dynamically at run time. In contrast, systems such as DIVE (Frécon, 2003) or MASSIVE (Greenhalgh, Purbick, & Snowdon, 2000) load all components each time they execute.

4. **Scalability**: How well can the DIA handle an increasing number of users? For example, although the DIVE architecture is very capable in certain areas such as concurrency, its use is limited by the fact that it is difficult to scale beyond about 32 participants (Carlsson & Hay-sand, 1993). Massively multiplayer online games have the ability to scale to thousands of simultaneous participants (Anarchy Online, 2004). Information management techniques such as dead reckoning and relevance filtering assist in making a DIA more scalable in addition to improving consistency. The design and deployment of the software is driven by the requirements of the DIA and the architecture of the underlying network (Singhal & Zyda, 1999). Most DIA architectures incorporate multithreading to improve the efficiency of the application and manage resources (Singhal & Zyda; Faisstnauer et al., 2000).
balancing, which distributes the workload efficiently among available resources, is used in
NetEffect (Das et al., 1997).

Snowdon, Greenhalgh, Benford, Bullock, and Brown (1996) reviewed distributed software architectures for
networked virtual reality, proposed a reference architecture for distributed virtual reality, and identified a num-
ber of key components which offer distinct services:

1. Security: authentication and authorization services;
2. Object support: memory management of persistent and executing object data;
3. Core virtual reality: services such as collision detection and interest management;
4. User interface: handles input and output from multiple devices.

These services are supported by additional services such as event management, database support, 3D
graphics rendering, and multicast network protocols.

The division of the software into components facilitates the development of new DIAs. For example the
continuum platform developed by Tran, Deslavgiers, Gerodolle, Hazard, and Rivierre (2002) is written en-
tirely in JAVA and is structured in layers, with orthogonal tasks such as object management and network sup-
port comprising the layers. These layers can be extended and adapted and they can be fitted together to derive
profiles, to construct different types of systems (e.g., a collaborative system and real-time online games). The
continuum platform is considered middleware as it provides services to the application.

Software deployment is also important. For example in SPLINE (Waters, Anderson, Barrus, et al., 1997a),
the software is spread across four different servers to improve scalability:

1. A session server to handle new connections;
2. A server to handle users with slow connections and interact with them using the client/server model;
3. A locale update server which is responsible for presenting participants with a seamless virtual envi-
   ronment from the various locales that it is divided into (Barrus et al., 1996a, 1996b);
4. A name server for the easy location of entities within the world.

A key component of SPLINE’s architecture is the locale update server. Koster (2000) examined the issue
of users congregating in a single locale and thus overloading a single locale update server. The solution pro-
posed by Koster is referred to as the Asheron’s Call architecture. This operates by dynamically distributing the
load among other available servers, resulting in a more scalable, more efficient, and more reliable system.

At the heart of a DIA is a database. This database is rendered to users in the form of a virtual world and
changes to the database must be rendered to interested users in real time. The existence of latency and the short
time limit imposed on rendering impose constraints on where the database and rendering software are located.
Communication across a network is usually impractical and so the DIA database is usually replicated at each
node or on a dedicated server for a LAN.

Software may also be deployed within the network itself. Kasera et al. (2000) proposed using active services
to efficiently deliver reliable multicast and hence increase the scalability of the DIA. By placing active ser-
VICES at strategic locations inside the network to control loss recovery and congestion, they demonstrated im-
proved resource usage, speedier loss recovery, and congestion control that respects TCP.

This section has given a brief overview of some of the issues that the software architecture must address and
that fundamentally impact the performance of the DIA and the user’s experience in the simulated world.

3. Concluding Remarks

This paper (Parts I and II) presented a comprehensive survey of the current state of research in tech-
niques for reducing and masking network latency and maintaining consistency in distributed interactive appli-
cations. It clarified some fundamental terminology and explored the concept of a DIA. It examined the issues
of consistency and latency, and offered a definition for each in the context of DIAs. The connection between
latency and consistency was explored. Finally, the paper reviewed the documented methods, mechanisms, and architectures available to the DIA designer in maintaining dynamic state consistency and reducing the impact of network latency.

An emerging approach to dealing with consistency and latency in DIAs exploits human perceptual abilities, user awareness, and patterns of events, in addition to how users interact with the technology to reduce the quantity of network traffic that needs to be communicated between users (Ryan & Sharkey, 1998; Stanney, Mourant, & Kennedy, 1998; Park & Kenyon, 1999; Ruhleder & Jordan, 1999; Vaghi, Greenhalgh, & Benford, 1999; Gutwin, 2001; Ruhleder & Jordan, 2001; Delaney, Meeneyghan, McLoone, & Ward, 2004; Gutwin et al., 2004; Mania, Adelstein, Ellis, & Hill, 2004; Wolff, Roberts, & Otto, 2004).

There are several other issues associated with DIAs that are not discussed in this paper. These include how new users should join the DIA (Roberts et al., 1995; Vogel, Mauve, Hilt, & Effelsberg, 2003), the exploitation of multimedia networking technologies (Gutwin et al., 2004) and the use of peer-to-peer networks such as Gnutella (Boukerche, Araujo, & Laffranchi, 2004). In addition there are recent efforts by the moving pictures expert group (MPEG) to extend MPEG to allow multiuser support (Joslin, Di Giacomo, and Magnenat-Thalmann, 2004). This extension to MPEG-4 is referred to as multiuser technology (MUTech). It remains to be seen how quickly these new standards and technologies will be incorporated into DIAs.

Acknowledgment

This material is based upon works supported by Enterprise Ireland under Grants no. SC/2002/129/ and IRCSET/SC/04/CS0289.

References


Macedonia, M. R., Zyda, M. J., Pratt, D. R., Brutzman,


