RESEARCH IN CONTINUITY OF OPERATIONS IN DISTRIBUTED TACTICAL INFORMATION SYSTEMS (U) PEGASUS SYSTEMS SUMMIT NJ D E PERRY 25 OCT 83 ARO-18622.1-EL-S
UNCLASSIFIED DAAG29-82-C-0014

END
**REPORT DOCUMENTATION PAGE**

<table>
<thead>
<tr>
<th>Field</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. REPORT NUMBER</td>
<td>A135320</td>
</tr>
<tr>
<td>2. GOVT ACCESSION NO.</td>
<td>A135320</td>
</tr>
<tr>
<td>3. RECIPIENT'S CATALOG NUMBER</td>
<td></td>
</tr>
<tr>
<td>5. TYPE OF REPORT &amp; PERIOD COVERED</td>
<td>Final 1 June 82 - 31 May 83</td>
</tr>
<tr>
<td>6. PERFORMING ORG. REPORT NUMBER</td>
<td></td>
</tr>
<tr>
<td>7. AUTHOR(s)</td>
<td>Dewayne E. Perry</td>
</tr>
<tr>
<td>8. CONTRACT OR GRANT NUMBER(s)</td>
<td>DAAG 29-82-C-0014</td>
</tr>
<tr>
<td>9. PERFORMING ORGANIZATION NAME AND ADDRESS</td>
<td>Pegasus Systems Summit, NJ 07901</td>
</tr>
<tr>
<td>10. PROGRAM ELEMENT PROJECT, TASK AREA &amp; WORK UNIT NUMBERS</td>
<td></td>
</tr>
<tr>
<td>11. CONTROLLING OFFICE NAME AND ADDRESS</td>
<td>U.S. Army Research Office Post Office Box 1221 Research Triangle Park, NC 27709</td>
</tr>
<tr>
<td>12. REPORT DATE</td>
<td>25 October 1983</td>
</tr>
<tr>
<td>13. NUMBER OF PAGES</td>
<td>15</td>
</tr>
<tr>
<td>14. MONITORING AGENCY NAME &amp; ADDRESS (if different from Controlling Office)</td>
<td></td>
</tr>
<tr>
<td>15. SECURITY CLASS. (of this report)</td>
<td>Unclassified</td>
</tr>
<tr>
<td>16a. DECLASSIFICATION/DOWNGRADING SCHEDULE</td>
<td></td>
</tr>
<tr>
<td>16b. DISTRIBUTION STATEMENT (of this report)</td>
<td>Approved for public release; distribution unlimited.</td>
</tr>
<tr>
<td>17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)</td>
<td></td>
</tr>
<tr>
<td>18. SUPPLEMENTARY NOTES</td>
<td>The view, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation</td>
</tr>
<tr>
<td>19. KEYWORDS (Continue on reverse side if necessary and identify by block number)</td>
<td>Distributed Systems, Recovery, Reconfiguration</td>
</tr>
<tr>
<td>20. ABSTRACT (Continue on reverse side if necessary and identify by block number)</td>
<td>This report is the final report for the first year of a project-ed 3 year effort on the continuity of operation in distributed information systems. We define a system model abstracted from a &quot;real&quot; battlefield information system, define simplifying assumptions to limit and focus the level of research, and present some preliminary results with respect to global system state recovery by means of an example common to recovery and reconfiguration.</td>
</tr>
</tbody>
</table>
Research in the Continuity of Operations
In Distributed Tactical Information Systems

Final Report

Dewayne E. Perry

25 October 1983

U. S. Army Research Office
Contract No. DAAG29-82-C-0014
Pegasus Systems

Approved for Public Release
Distribution Unlimited
The Views, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as official Department of the Army position, policy, or decision, unless so designated by other documentation.
1. Introduction 1
2. Research Goals 1
3. Summary of Results 2
4. Research Results 2
  4.1. The System Model 2
  4.2. Recovery 5
  4.3. Summary 8
1. Introduction

This final report represents the first year effort in a projected three year research effort into the continuation of operation in distributed tactical information systems. The principle investigator is Dr. Dewayne F. Perry of Pegasus Systems and the work is sponsored by the Army Research Office under contract DAAG29-82-C-0014 for the period 1 June, 1982 through 31 May 1983.

2. Research Goals

In this research project, we are investigating the techniques and tools required to build reliable systems that provide continuity of operations despite component failure. Specifically, we are interested in software solutions to the problems of reliability, recovery and reconfiguration.

The basic goals of this research, over the three year effort are summarized as follows:

- To identify and refine the fundamental issues and the relationships between these issues in the construction or survivable distributed systems;
- To survey existing approaches to these various issues, to determine which are suitable for survivable systems, and to explore their interaction with each other;
- To determine in levels of performance by these various design and implementation strategies; and
- To establish and design approach applicable to survivable distributed tactical information systems.

The basic underlying issue is the extent to which the goal of survivability places special requirements upon the various aspects of system design and construction. Are there existing techniques that are satisfactory within the constraints of acceptable system performance or must new techniques be developed to solve the particular needs of continuous operation?

The goals of this research is explored through the use of one or more models of actual tactical information systems. These models are designed in various stages in order to fully understand the various solution strategies, their implications and their cost. While these models represent "real" systems the emphases is upon computer science issues rather than on application related issues.

The Ada language will be used to specify, design and construct these various system models in sufficient depth to expose the relevant problems of the design and, where they occur, the limitations of Ada to handle the problems of distributed system design and construction.

The goal for this first year of effort are as follows:

- define a model system abstract from a "real" field situation;
within the context of this model, identify the simplifying assumption are to be made in order to restrict the complexity of the problem and control the interaction between various issues; and

- address the issues of redundancy, focus on control and reconfiguration.

The effort with respect to the third goal has centered primarily on issues of recovery.

3. Summary of Results

First, the system model has been defined as an abstraction or a battlefield information system: a set of battalions with a set of forward observers. This model has all the requirements needed for a framework for considering the survivability of a distributed information system:

- distributed processors (the battalions of the model);
- distributed resources (the forward observers); and
- distributed information (the battalion database).

A description of the model is to be found below.

Second, the simplifying assumptions have been delineated that restrict the focus of the investigation to the appropriate level of detail and that support the issues under investigation. These assumptions have been made in order to concentrate our attention on the problems of recovery and reconfiguration, in response to component failure. These assumptions are presented with the system model in the discussion below.

Finally, we have focused our attention specifically in the problem of global system recovery and present a methodology for determining, and recovering the system to, the most recent consistent global state. An example common to recovery and reconfiguration, that of transferring databases for backup and repartitioning purposes, is given to illustrate the methodology.

4. Research Results

The results of this research are discussed in the next two subsections. First, we present the system model and accompanying simplifying assumptions. Second, we discuss the problems of global recovery and present a methodology for determining global recovery states in distributed, cooperating processes.

4.1. The System Model

The system model described below is abstracted from a "real" military system: a set of battalions together with a set of forward observers which serve as the input sources; the output of the systems can be ignored with loss of generality. The battalions are independent components, dispersed geographically, that cooperate...
with each other in gathering tactical information and that pursue individual as well as global goals. The forward observers serve as the source of the tactical information.

By abstracting away the application dependent aspects of the system, we define the following system model for our research. The model consists of:

- a set of autonomous, homogeneous nodes,
- a set of remote input sources,
- a database of information, and
- a fully connected network.

We will discuss the simplifying assumptions with results to these model concerning the following issues:

- node behavior
- remote input behavior
- database characteristics
- communications
- local storage
- redundancy
- recovery

The nodes in the system may enter and leave the system. It may be useful to distinguish two kinds of node removal from the system: planned and unplanned. In the first case, the system can a priori provide the temporary coverage within the system for that node (i.e., a basic minimum reconfiguration that allows for node reentry later). In the second case, the worst case is assumed and the system recovers by reconfiguring around the missing node. Whether reentry into the system causes reconfiguration is application dependent—we assume it does cause reconfiguration.

Remote input sources behave in a manner similar to that of the nodes—they enter and leave the system. The system provides a mapping of remote input sources into nodes that is reconfigurable dependent upon the behavior of the nodes and of the remote input sources. Remote input sources report to the system and the system provides the mapping dynamically.

We assume that there neither malicious nodes nor malicious remote input sources. Further, we assume that
each node detects its own failure and recovers to a consistent state. Local recovery - that is, without reconfiguration - maybe possible for this state of affairs.

The basic goal of the system with regard to the data base is no loss of data or information. For the sake of simplicity, we assume that the data base is partitioned. Each node is responsible for its own data base. Further, the mapping of remote input is not done to nodes per se, but to the partitioning of the data base - and set of remote input sources is mapped to a particular partition of the database. Replication of the data base is considered from the perspective of backup purpose only. The rationale for this approach is that we are not concerned with accessing the data but only with maintaining the completeness of the data. Each partition then is replicated on some other node.

It is in the area of communications that we make the stringent assumptions because we wish to concentrate on the higher level issues rather than on the mechanics of the connection between components. First, we assume that the communication is fault free - either the communication completes correctly, or it does not complete at all (and this latter condition occurs only if the destination node is unreachable). Second, the network is adaptive in such a way that all nodes agree as to the faults in the system and as to the nodes connected to the system. Third, notification occurs when a node or set of nodes is not reachable. Finally, we assume that packet radio is used in order to provide the basis for performance determinations.

Local storage for each node is assumed to be stable in the sense of [Lampson 76]. As a result of stable storage and fault-free communication we can assume there are no consistency problems with respect to the data. The only problems are those due to latency of data between nodes.

As previously mentioned, the data base is fully replicated by means of a single copy backup. In order to focus our attention on the most basic interaction, we make the (extremely simplifying) assumption that the partitioning of the data base is such that the source and backup copies will not both be lost in a single recovery/reconfiguration cycle.

For recovery, we assume that only one node will disappear at a time and that there is sufficient time between disappearances for the recovery and reconfiguration to take place. Later, we will relax this assumption. Further, we assume that a director of recovery and reconfiguration is selected to direct the repartitioning and reconfiguration of the system.

With these assumptions, we have the basic framework to investigate the issues of recovery at a relatively high level in order to examine the interactions between nodes.
4.2. Recovery

Some of the basic problems that occur in a distributed system that result in changes to the system, or global state, are as follows:

- a node fails,
- a node enters the system,
- the interconnection between nodes fails
- the system is partitioned

The traditional notion of recovery - returning to a previously consistent state - is not entirely applicable here. It may be impossible to return the system state to a previous condition. The error conditions or the causes of the error conditions may preclude such a return. It is the very disintegration of the global state that may be the cause of the problem in a distributed system. Particularly since the connections between components and the components themselves that comprise the global state may fail.

It is for this reason - the disintegration of the global state - that backward recovery may not be a viable recovery mechanism and that we turn our attention to forward recovery considerations. Recovery in a distributed system under these conditions is more a matter of reconfiguration and the establishing of a new system state. The basic goal, however, is still to maintain a consistent global state with regard to the data base.

A basic part of the recovery by reconfiguration is concerned with repartitioning the database. When a node fails, the data maintained by the failed node needs to be integrated into another node and the remote input sources remapped to that node. We will assume for the time being that the partition is kept intact, primarily because repartitioning multiplies the amount of work for reconfiguration and we want to look at the simplest case first.

One of the major concerns in this reconfiguration is the movement of a new backup copy of the database to the new backup node. This involves the participation of three nodes: the initiator node, i.e., the director of the recovery; the source node of the backup file; and the destination node where the partition is now to reside, either as a backup or as integrated into the nodes database. The following represents the interaction between the nodes.

```
Initiator       connect to       Source
Initiator       -->             Source                           : send database to dest
Source           connect to     Destination
```
The interaction of the three nodes can be seen in the following state description of the three nodes simultaneously. Where the states proceed concurrently, the nodes can proceed concurrently. Where there is no processing indicated, the node is waiting for synchronization on the next non-empty global state. The global states are separated by lines; global state transitions that represent global recovery points are indicated by asterisks.

<table>
<thead>
<tr>
<th>state</th>
<th>Initiator</th>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>decide recovery plan</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>connect to S wait for accept</td>
<td>wait for connect accept connect</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>send request to S</td>
<td>receive request</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>connect to D wait for accept accept connect</td>
<td>wait for connect</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>send request to D</td>
<td>receive request</td>
</tr>
<tr>
<td>---</td>
<td>----</td>
<td>-------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>---</td>
<td>----</td>
<td>-------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>prepare database</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>----</td>
<td>-------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>send database</td>
<td>receive database</td>
</tr>
<tr>
<td>---</td>
<td>----</td>
<td>-------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td></td>
<td>install database</td>
</tr>
<tr>
<td>---</td>
<td>----</td>
<td>-------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>receive installed</td>
<td>send installed</td>
</tr>
<tr>
<td>---</td>
<td>----</td>
<td>-------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>disconnect from D</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>----</td>
<td>-------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>receive db installed</td>
<td>send db installed</td>
</tr>
<tr>
<td>---</td>
<td>----</td>
<td>-------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>disconnect from S</td>
<td></td>
</tr>
</tbody>
</table>

In the normal case, the global state transition occurs in parallel with interacting processes proceeding concurrently on the basis of the message transmission and reception. A send does not complete until the message has been received. Non-interaction processes proceed directly to their next non-empty global state.

If a connection is broken and then resumed, the two interacting processes must determine the the most recent state to reinstate. The usual case is to recover back to the earliest of the two processes state and proceed from there. In this case, however, one of the processes may be able to establish a more recent state as the effective state because it represents the system's global state where it is most important - in this case, with respect to the database.

When connections are broken, and connection is re-established, there are several possible courses of action.
First, the recovery of the global state may only affect the two processes involved. In this case, no further recovery is needed but that needed to resolve the appropriate state of the system. Second, the recovery may affect other processes involved in the operation and thus a more global recovery action is needed.

For example, once state 4 has been reached, there is no need to recover the cooperating processes to a point earlier that that if the connection between the Source and The Destination is broken unless the Source node must recover to a state prior to state 4. If the Source node fails and recovers to global state 8 but the destination process is state 10, there is no need for the both the Source and the Destination node to recover back to state 8 when state 10 will maintain the consistency of the recovery plan - it already has installed the database.

The net result of this kind of analysis is that recovery can move forward to a later state (that is the most recent global recover state), rather than back to an earlier state, under certain conditions.

In summary, the following considerations result from this methodological approach.

- There are identifiable global states that represent either the completion of interaction between two cooperating processes (or nodes) or that represent important events in one of the processes that can be considered the point event (for example, the reception of the data base, or the installation of the database).

- An important question is how to represent and distinguish local and global states. For example, a group of local states might be grouped together to represent an important global state with respect to the other cooperating processes.

- Synchronization points are expressed by message exchanges and represent important transitions in the global state.

- The dichotomy between local and global state changes can be expressed in parallel for the cooperating processes. Even they are distinct events, they do occur in parallel and can be represented that way.

4.3. Summary

The preceding discussion represents a preliminary investigation into the problems of recovery in distributed systems. A system model has been constructed and restrictions placed on it to narrow the focus of the research. The approach to determining forward recovery points is an initial look at ways of reconstructing the global state of a distributed system when the global state disintegrates because of node failure.
References

Recovery Blocks in Action.
1976.

A Model of Recoverability in Multilevel Systems.

System Fault Tolerance.

The Provision of Recoverable Interfaces.

*System D: a Distributed System for Availability*.

*SR: A Language for Distributed Programming*.
Department of Computer Science.

Software Reliability in Real-Time Systems.

Recover Scenario for a DB/DC system.

*System R: an Architectural Update*.


Distributed Reconfiguration Strategies for Fault-Tolerant Multiprocessor Systems.

Optimization of the Number of Copies in a Distributed Data Base.
Recovery Semantics for a DB/DC System.

[Denning 76] P. J. Denning.
Fault-Tolerant Operating Systems.

[Durham 81] Ivor Durham.
The Design and Construction of Reliable Software for Distributed Computing.
March 1981.

[Garcia-Molina 82]
H. Garcia-Molina.
Using Semantic Knowledge for Transaction Processing in a Distributed Database.
June, 1982.
Department of Electrical Engineering and Computer Science.

[Gracia-Molina 80]
H. Garcia-Molina.
Reliability Issues for Completely Replicated Distributed Databases.
In CompCon 1980, pages 442-449. IEEE, Fall, 1980.

Notes on Data Base Operating Systems.

The Recovery Manager of a Data Management System.

[Hecht 76] H. Hecht.
Fault-Tolerant Software for Real-Time Applications.

Decentralized Control.


A Systematic Design and Analysis of Reconfigurable Distributed Computer Systems.

The Implementation of Reliable Distributed Multiprocess Systems.
_Crash Recovery in a Distributed Data Storage System._  

An Analysis of Different Approaches to Distributed Computing.  
In _First International Conference on Distributed Computing Systems_, pages 222-232.  
Huntsville, Alabama, October, 1979.

A Distributed System for Real-Time Transaction Processing.  
In _14th Hawaii International Conference on System Science_. Honolulu, Hawaii, January, 1981.

_Program Structures for Exceptional Conditional Handling._  

[Linden 76] T. A. Linden.  
Operating System Structures to Support Security and Reliable Software.  

[Lindsay 79] B. G. Lindsay, et al.  
_Notes on Distributed Databases._  

Primitives for Distributed Computing.  

Checkpointing and Error Recovery in Distributed Systems.  
April, 1981.

_Medusa. A Distributed Operating System._  

[Randell 75] B. Randell.  
System Structure for Software Fault Tolerance.  


[Randell 79a] B. Randell.  
System Reliability and Structuring.  
[Randell 79b] B. Randell.
Software Fault Tolerance.

[Ryan 80] T. M. Ryan.
Backup and Recovery for Distributed Interactive Computer Systems.

*Distributed Computer Systems: Structure and Semantics.*

Reliability Issues in Distributed Information Processing Systems.
