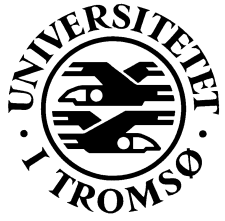


REPORT

Computer Science Technical Report: **94-18** **December 1994**

Satellite Communication Applied in a Distributed Application

W.Farstad, D.Johansen, G.Hartvigsen

INSTITUTE OF MATHEMATICAL AND PHYSICAL SCIENCES
Department of Computer Science
University of Tromsø, N-9037 Tromsø, Norway, Telephone +47 77 64 40 41, Telefax +47 77 64 45 80

Satellite Communication Applied in a Distributed Application

W. Farstad, D. Johansen, G. Hartvigsen

Department of Computer Science,
Institute of Mathematical and Physical Sciences,
University of Tromsø,
N-9037 Tromsø, Norway.

Abstract

The paper describes the development of a prototype application for access to vital weather information from the Northern Atlantic sea region. The application gives meteorologists access to weather observations measured on sea vessels. Today the available information is very limited. On a daily basis, only 4-5 weather observations in the whole arctic sea-region are conducted. It is therefore suggested to use the available fishing-float and the coast-guard as a basis for a full scale application. Communication is based on the Inmarsat-C global satellite system. This kind of communication offers limited data bandwidth, currently 600 bit/sec. In addition delays are introduced by the coast/earth-station due to message-queues and protocol-transformations. Based on the services offered by the Inmarsat-C system, a simple prototype communication mechanism for data communication between terrestrial computers and computers on sea-vessels has been developed. It is, partly due to the nature of the satellite system, based on unreliable multicast of messages. A message may consist of a request which requires a reply (multicast RPC). Test results showed that doing a request and getting a reply takes from 1.5 - 4 minutes. The results depend on the type of message (supported by the Inmarsat-C system) which is used. The sea-vessels send weather observations in set intervals. This means higher availability on weather observations, and again a big step for meteorologists which today have to rely on observation which may be several hours old (if any at all).

1 Introduction

This report is part of the StormCast research project [Hartvigsen88][Johansen91] [Johansen94]. The project has been under continuous development since 1988 and is a project with both meteorological and environmental aspects. The StormCast project's goal is to develope an architecture for construction of distributed monitoring applications. The domain for these application is mainly weather- and environment. In this report the meteorological aspect is addressed.

Weather observations from the Northern Atlantic sea region are currently reported from sea-vessels in reduced amount and frequencies. The reporting is based on manual observations and transmission via radio calling or telex. The transmission of observations via telex is timeconsuming. Typically, it takes hours from the observation was done until the meteorologist has the data on his desk. Parameters of interest is typically wind-speed, wind-direction, air-temperature, relative humidity etc. One of the main reason for the need for more observations from these areas is polar lows. This is small and intense low-pressures which arise at the polar ice-front and from there moves rapidly towards the south. Such polar low-pressures may have caused many shipwrecks.

By placing automatic logger stations and equipment for satellite communication on a group of sea-vessels, the weather status from these vessels can be reported near to real-time. This offers the meteorologist a much better fundament for their prognoses and weather reports. This will increase the reliability of the prognoses for the potential users which may be the same who did the observations.

A key point of the work presented in this paper is to increase the availability of weather observation from the sea regions. This has been done by the development of a simple communication protocol based on the services that the global satellite system, Inmarsat-C, today offers. In this project the earth-station at Eik in the Rogaland county in Norway is used as a link between the terrestrial and the satellite network. The use of one earth station is only a practical constraint. In order to test the protocol, we developed a prototype distributed monitoring application. The applications main purpose is to offer meteorologists more available and accurate observations. This is accomplished by offer more data and also faster than what is available today.

The rest of this paper is structured as follows. Chapter 2 describe the StormCast project. Chapter 3 discusses satellite communication and Inmarsat-C. In chapter 4 we construct a simple protocol and a prototype application. In chapter 5 test results are presented. Chapter 6 discusses the developed application and the test results. Chapter 7 concludes the report.

2 The StormCast Project

The motivation for the StormCast project [Hartvigsen88][Johansen91] was the need for prototypes on distributed applications to examine distributed operating systems as the Amoeba [Tanenbaum90] distributed system. Consequently, development of a software architecture has been a central part of the project. The architecture is a fundament on which meteorological and environmental monitoring distributed application is to be built. The architecture

covers all areas of such applications, from monitoring and collection to data storage and visualization. The architecture also addresses common computer science problems in the field of distributed systems, such as increased availability and reliability.

The current focus in the StormCast project is to develope a meteorological workbench [Johansen1993a/b][Hartvigsen1994]. The StormCast meteorological workbench will among others provide a tool for visual observation through the use of site cameras, distributed artificial intelligence applications for wind forecasting, monitoring of weather and environmental data, visualization of weather forecasts (from numerical models) and different multimedia data (including video, audio and satellite pictures). The architecture also addresses mobility. The StormCast architecture consists of six layers (see Figure 1)..

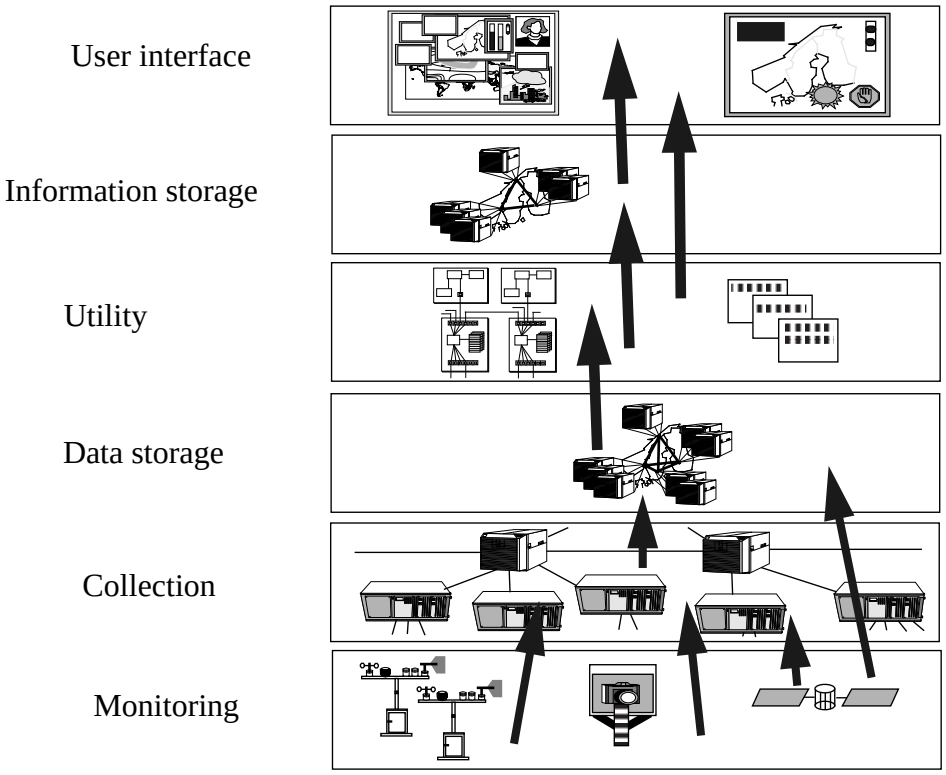


Figure 1. The StormCast 2.1 architecture

The monitoring layer is typically represented by logger stations with sensors that measure physical values like temperature, wind speed, air humidity, air pressure i.e and represent these values in a digital, computer readable, form. The services this layer offer is reliable (by replication of physical instruments) but could have reduced availability caused by the communication used between the collector and the logger station (i.e. satellite communication).

The collection layer’s task is to collect data from geographically different areas to compensate for vulnerable communication at the bottom layer. Increased availability is achieved by caching techniques at this level. The data storage layer stores measured data (raw data) permanently. This raw data comes from the monitoring or the collection layer.

The application layer consists of services which do computations on the raw data from the layers below. Processed data may either be stored at the information layer or be presented at the user interface layer. The user interface layer implements the user dialog with the application. At this layer a unit called a Service Provider [Johansen93a/b] is introduced. This is meant to be a front end for the user interface for available services (Figure 2).

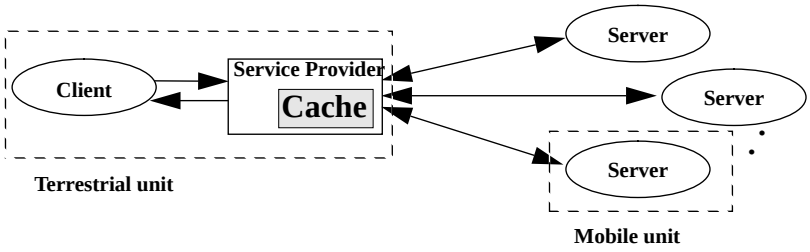


Figure 2. Service Provider

A Service Provider's main task is to receive and handle requests from clients. The service provider caches all replies and uses these cached data to handle requests. In this way the client achieves higher availability and performance. When the servers are available via satellites or modems (low-bandwidth communication), the extra cost to check the cache instead of contacting the servers directly, is not assumed to represent a reduction in performance. Further, the service provider is used to compensate for controlled and uncontrolled disconnections. Compensation for uncontrolled disconnections is done by monitoring the user-interaction pattern. Based on this the service provider can update itself with the most interesting data. This means that client requests are ideally always answered with cached information. Controlled disconnection is done when it is known in advance when the services becomes unavailable. This could be done after model from Codas hoardwalk [Kistler91].

The StormCast model is based on communicating distributed modules, where a module may be a process (in the Unix sense), a stand alone computer (PC) or a logger station (a logger station may as well be represented by a process). By distribution we mean geographical dispersion where only the physical communication is a limiting factor. The communication between modules is based on a client-server interaction model. Hence, services are typically carried out bye one or several servers upon requests from a client in a higher layer. A group communication model is also used, for instance for replicas of software modules at the same layer providing a more fault-tolerant service than a single module can do.

3 Satellite Communication and Inmarsat-C

Satellites in geostationary orbits, i.e. 36000 km above the equatorial earth plane, have the ability to cover up to 25% of the earths surface. A satellite system is of nature a broadcast medium. Later we will see how this is utilized by constructing a simple multicast RPC communication protocol. The Inmarsat-C description is from [Enersen89], [Hanebrekke90], [Berg90] and [Inmarsat93].

Inmarsat-C offers global data communication between users in the terrestrial data network (i.e. X-25) and mobile Inmarsat-C terminals. The Inmarsat-C system consists of tree main components: the satellites, mobile earth stations (LES) and land earth stations (LES and NCS). Currently the system consists of four satellites in geostationary orbit covering the earths surface from 75 degrees north to 75 degrees south. In addition spare satellites are available in case of errors. Each satellite covers approximately one fourth of the earths surface, and is placed over the four sea-continents: Atlantic west, Atlantic-East, Indian ocean and the Pacific ocean. We used the satellite over the Indian ocean.

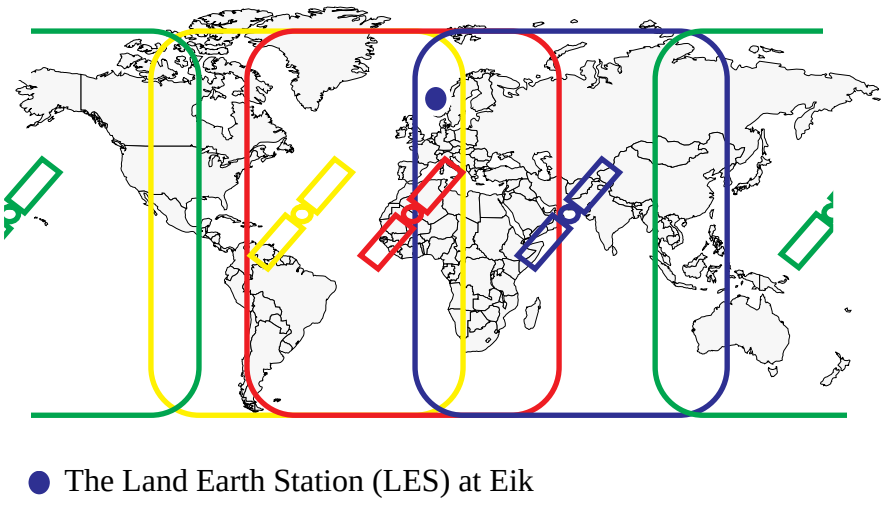


Figure 3. Inmarsat-C coverage areas

Figure 3 shows the Inmarsat-C coverage areas. These satellites has the ability to communicate with mobile units which are inside the coverage areas. Communication between mobile units and terrestrial units passes trough a land earth station. This can be seen as a gateway between the satellite network and the terrestrial X-25 network. Figure 5 on page 7 illustrates the integration of these networks. Here only one of the satellites with its NCS and to land stations are shown. In addition tree mobile earth stations are also shown. Solid lines between the units show which links/channels interconnecting the stations (all these connections go via the satellite). The three different types of earth stations are: LES (land earth station), MES (mobile earth station) and NCS (network coordination station). The NCS's responsibility is to coordinate all traffic between LES and MES. It is one NCS per coverage area. The NCS therefore has to be updated with which MES is in its area, which LES and MES is in use and which resources are available in its area. The LES main responsibility is to interconnect the Inmarsat-C network with available terrestrial networks (i.e. X-25). There could be several LESs per satellite. One LES could also handle several satellites (i.e. the LES at Goonhilly in Great Britain).

A MES consists of an antenna, a satellite modem and a terminal, usually a ordinary personal computer. Figure 4 on page 6 shows a common solution for a MES.

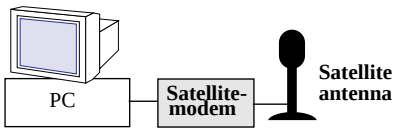


Figure 4. Mobile earth station configuration

Currently the Inmarsat-C system offers a data rate of 600 bit/second, expected to be raised to 1200 bit/second in the next generation. The relative low data rate is a consequence of the requirement of a small and cheap satellite transceiver. A satellite modem and antenna cost approximately $5000-$6000.

The Inmarsat-C system is currently a relatively high-cost service. The costs are calculated based on the number of bits transferred. Messages are split into segments of 256 bit, and a segment (March 1994) costs approximately 30 cents.

3.1 Disconnected operation

As Figure 3 on page 5 illustrates the availability is limited to the area between 75 degrees north and 75 degrees south. When ships (mobile units) are beyond 75 degrees north (or south) there will be problems with the communication. It is, however, shown that communication could be available even beyond 75 degrees by using several antennas at different heights [Hanebrekke90]. Availability is a key issue here. We say that mobile units are unavailable if:

1. they are outside the coverage area.
2. they are inside the coverage area, but the communication is malfunctioning.

The mobile unit and the terrestrial unit must therefore compensate for disconnected operation. We define *disconnected operation* as a state defined in 1) or 2).

Some of the ideas from Coda [Kistler91] may be transferred to this project. In StormCast we only operates on immutable data [Johansen93]. This means that disconnection does not raise consistence problems. Specially the *hoardwalk* function in Coda, which compensates for uncontrolled disconnections, is of interest. It is possible to monitor the user interaction, and use this to keep the service provider up to date with the “most interesting” data. Also the function which makes it possible to do controlled disconnections in Coda is interesting. This means we can do a controlled disconnection by first contacting all interesting (either from a monitored pattern or from a user profile) logger stations and download the last observation. The type of information we operate on (weather observations) will be less interesting as time pass by. Requests about the latest observation from a certain area will therefore be less relevant as time pass by.

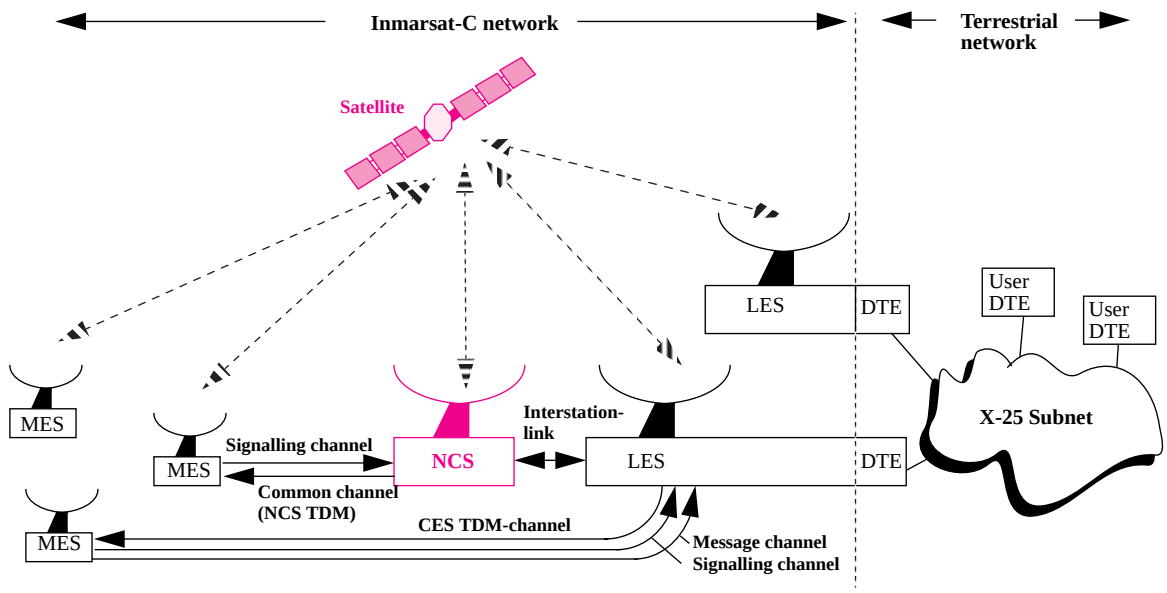


Figure 5. Integration of the satellite and the terrestrial X-25 network.

3.2 Communication and naming

3.2.1 Inmarsat-C communication

Messages from the terrestrial subscriber can be sent by one of the following methods:

1. Ordinary message (ASCII text).
2. Poll (binary data).
3. Enhanced Group Call (EGC), (ASCII text).

The terrestrial subscriber may receive the following types of messages:

1. Ordinary message
2. Data report

Messages from a mobile unit may be sent as:

1. Ordinary message
2. Data report

The mobile units may receive the following:

1. Ordinary message
2. Poll
3. EGC

Exchange of messages between the mobile units and the terrestrial subscriber is based on store-and-forward of messsages.Transmission of ordinary messages requires establishment of a connection before the data transmission is done. There is no limit on the size of data to transmit. Polling and data reports are short binary data messages that is transferred on the signalling channel. With polling and data reporting the overhead of connection establishment is avoided, resulting in higher performance for small messages (poll: ≤ 300 byte, data report ≤ 32 byte). A poll could be sent from a terrestrial subscriber and a data report could be sent from a mobile unit. A data report is typically a reply from a poll-request. This service is therefore well suited for monitoring and surveillance.

Enhanced group call (EGC) is a service for sending calls to a group of mobile units. A group is either characterized as a fleet or an area. The EGC service is therefore split into FLEETNET and SAFETYNET. The FLEETNET service is used to send messages to a group of mobile units belonging to a organization. With SAFETYNET it is possible to send messages to all mobile units in a geographically limited area.

Ordinary messages and poll could also be addressed to a group of mobile units. This method will be used in this report. A poll is addressed to a group of mobile units with a group address (called a DNID, Data Network IDentifier).

3.2.2 The land earth station (LES) at Eik

The various LES in different countries have different computer interfaces and services. The practical issue in this report is based on the services that the Eik LES today offers. In [Inmarsat93] the services are described. Due to the differences in the interfaces and some of the services it is preferable to develope a standard communication protocol that is LES independent. This will simplify the development of a distributed logger system. The development of such a protocol is done by enhance and integrate the services that the Inmarsat-C system today offer. For the terrestrial unit this means to utilize the X-25 communication interface that the Eik LES offers (most of the LESs' use the X25 protocol as its data communication interface) [Inmarsat93][ABBnera93]. For the mobile unit it means to utilize the services that the specific satellite modem offer (here we used the Trane & Trane 3020A transceiver [T&T92], see Figure 6)..

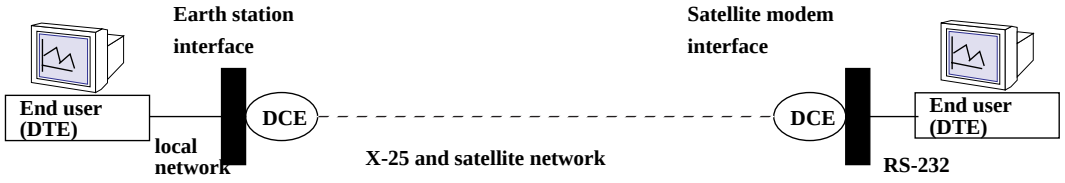


Figure 6. Interfaces that end-users sees

The earth station interface is based on a interactive user interface which is also possible to interact with programmatically. This is done by sending X-25 packages filled with commands and messages which instruct the earth station what to do. A set of X-25 packages could, for instance, contain commands and information telling the earth station to send a poll to a group of mobile units. This interface is specified as a data protocol, Terrestrial Data Pro-

tocol, in [ABBnera93]. This constitutes the terrestrial units interface to the satellite network. All messages that is received at the LES is not forwarded until the whole message is received from the sender – both the terrestrial and the mobile unit (store-and-forward based).

3.2.3 Naming

Naming is addressed to ensure proper scaling of the distributed logger system. The application must be independent of the number of mobile units (loggers) that is part of the application. One of the goals with naming in distributed systems is that the units shall be independent of physical connection, topology and location. This raise the need for three types of identifiers: name, address and route. As argued by [Scoch78], “a resource’s name indicate what we search, an address indicates where it is and a route tells how to get there”.

A name can either be mapped directly to a route or to an address, which again is mapped to a route. As for the OSI-RM, names is mapped to addresses at the application layer. Here a directory service could be used. Address to route mapping is often done at the network layer where net specific routers are used. Consequently, it is more common to map names to addresses than directly to a route [Patel90].

In this report we have constructed a simple directory service that maps mobile unit names to subnet addresses (X-121 and Inmarsat-C addresses). This will be described in the following section.

4 Construction of a simple protocol and a prototype application

4.1 Application requirements

Below the most important application requirements are presented. These are based on cooperation and discussion with typical end users, i.e. meteorologists. The main task for this application is to *offer end users current weather observations in addition to historical data from the northern arctic sea region*.

The application have to compensate for disconnected operation. This is accomplished by structuring the application as shown in Figure 2 on page 4. Here we call the service provider “data storage unit”. The servers on mobile units we call “collection units” and the client is called “visualization unit”. Notice that the collection unit (layer 1 and 2 in the StormCast architecture) is placed on the mobile unit (i.e. a PC) and the data storage- and visualization are both placed in a Unix environment (workstation in a Ethernet LAN).

4.2 Construction of a communication protocol

Based on the requirements two protocols will be developed. The data storage unit (terrestrial unit) must be able to send a message to all the collection units (mobile units). The collection units also must be able to send observations to the data storage unit (or units). This can be viewed as an unreliable multicast protocol.

In addition there must be possible to send a request to a group of collection-units asking for the current observation. This can be viewed as a multicast RPC protocol [Satyanarayanan90c].

4.2.1 Multicast protocol

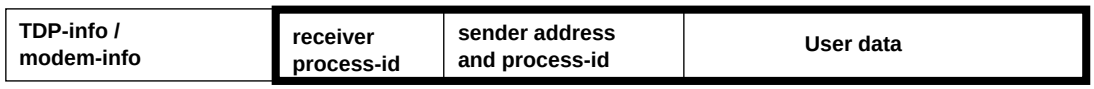
This protocol is a unreliable datagram service. There is no guaranties that a message is delivered to its destination. The protocol is an intermediate between a UDP [Stevens90] kind of protocol and a message passing protocol. The sender must be able to address a group of receivers. The protocol primitives are shown below (in C syntax).

```
void sat_bind(proc_id, proc_name)

int sat_sndmsg(proc_id, sender_address, group_address |
               reciever_address[es], u_data)

void sat_rcvmsg(proc_id, sender_address, u_data)
```

Messages sent, i.e. that `sat_sndmsg()` builds, are structured as illustrated in Figure 7.



Inmarsat-C dependent part. Protocol dependent part (max 1200 byte).

Figure 7. Protocol message

A message like this could be addressed to:

- a single receiver
- a list of receivers
- a group identifier

This message could be sent either as a poll (addressed by DNID), a data report (addressed by DNID) or as a ordinary message (addressed by a single address or list of addresses). Polling and data reporting are used when the user data part is small ($\leq$ 300 byte for poll and $\leq$ 32 byte for a data report). If this message is to be used for multicast all receivers must apply the same process-id. Messages does not contain sequence numbers, since the protocol does not handle sequencing. A message is therefore a complete dialog unit.

`sat_bind()`

This primitive returns a free process-id. This is used for naming of the communicating processes.

`sat_sndmsg()`

Parameters to this primitive are process id returned from `sat_bind()`, sender, receiver(s) and user data. On the terrestrial side the function assemble the message in the required number of X-25 frames. On the mobile unit side the function assemble the message and deliver it

to the transceiver and thereafter commands it to send it. The function returns the result of the delivery, i.e. if the land earth station received the message.

`sat_rcvmsg()`

Parameters here are process-id, sender and user data. This function blocks on the given process id. When a message addressed to this process is coming in, the process is freed. On the terrestrial side X-25 packets are assembled to a complete message (max 1200 byte). This is then delivered to the blocking process. On the mobile unit side the message are retrieved from the transceiver and are delivered to the receiving process.

`sat_sndmsg()` may be used to send a message to many receivers, while `sat_rcvmsg()` returns when a single message are received. To get a request-reply mechanism a multicast RPC-mechanism are developed based on the multicast protocol in the next section.

4.2.2 Multicast RPC protocol

The protocol is based on Amoeba's syntax for remote operations [Tanenbaum90]. Remote procedure calls are implemented by using this mechanism trough stub-functions. This is not a traditional RPC-mechanism [Birrel84]. Traditionally it is transparent where the remote procedure is localized. This means that there are just one instance of this procedure (assuming no replication) available. In this context the same procedure are available on many machines (mobile units) where the criteria for which of them to use are based on location. Address to which mobile unit to contact must therefore be a parameter to the RPC stub function. [Goscinski91] define tree forms of RPC semantic:

- maybe once
- at least once
- exactly once

Due to the physical limitations in the satellite system used, the RPC semantic here will be "maybe once". This means that the request and the reply may be lost. It is not possible to tell whether the operation was performed. This is a adequate situation for the purpose of fetching weather data.

4.2.3 The protocol

By building on the primitives from the last section we develope a simple RPC protocol. The protocol primitives are shown below (in C syntax):

```
int put_request(receiver_address[es], sender_address, u_data,
                proc_name, nbr_reply)

void get_request(sender_address, u_data)

int put_reply(receiver_address, sender_address, u_data)
```

A client using `put_request()` trough a stub function names the receiving group, itself, the message, the name of the receiving process and the number of replies expected. The name of the remote procedure and parameters are given in the message (`u_data`).

`put_request()` is used by the client-stub and blocks until all in the group has answered or a timer has elapsed (time based on the test results). The stub must allocate memory for the reply. The function returns with the information it get in this time. Data that is returned from any server will be in the form of a sender address and the result from the remote procedure. This is assembled to a message which the caller of `put_request()` must handle. Figure 8 on page 12 is an example of a stub function.

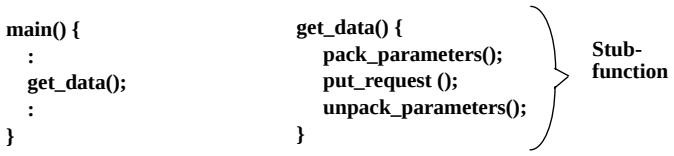


Figure 8. Stub function

On the server side it is assumed that `get_request()` is executed. This is a blocking call that the server stub apply. When this is released the stub disassemble the message, calls the actual function, get an answer which are assembled. This is sent back with `put_reply()`. Figure 9 illustrates the protocol.

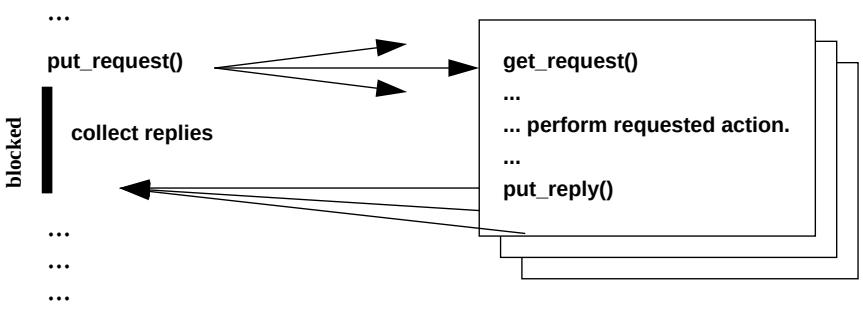


Figure 9. Multicast RPC-protocol

The user data field in the multicast protocol are split in two; stub procedure name and its parameters. This is illustrated in Figure 10. Return messages are also placed in the user data field.

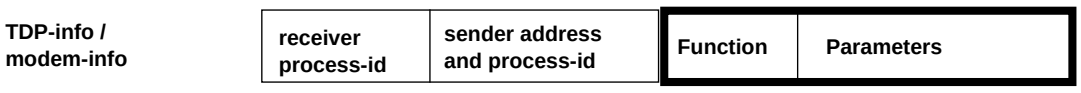


Figure 10. Message for multicast RPC request

The next figure illustrates a summary of Section 4.2 on page 9 related to the OSI reference model. The protocols are on the terrestrial side based on the TDP protocol and the LES at

Eik. At the mobile unit side the protocol are based on a transceiver of the type Trane & Trane 3020A.

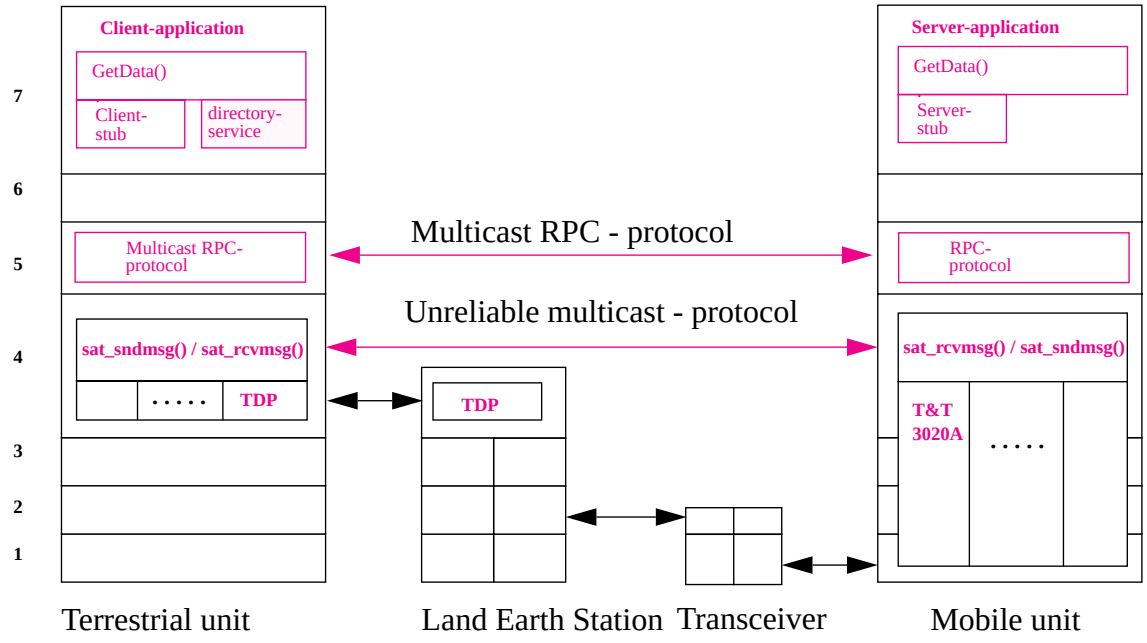


Figure 11. The communication mechanism

4.3 Construction of a StormCast application

Figure 12 on page 14 illustrates the overall design: the visualization unit, the data storage unit and the collection unit relative to the StormCast architecture. At the data storage level this is a centralized solution. Transmission of weather data is done on initiative from the data storage unit or from the collection unit. Like Coda [Kistler91] collection units are seen as primarily available. There are two possibilities for collection units being unavailable: 1) communication failure, and 2) the units are outside the Inmarsat-C coverage areas. The pre-sumptions are that the mobile stations are logged on the satellite system and the collection units are started.

The application therefore must be aware of the possibility of disconnected collection-units. The application compensate for disconnected operation by:

- the data storage unit receive weather data in (user) set intervals. Compensates for uncon-trolled disconnections.
- the user may send a multicast message to all collection units to get the last data. Used for controlled disconnections.
- the collection unit cache data when it operates disconnected.

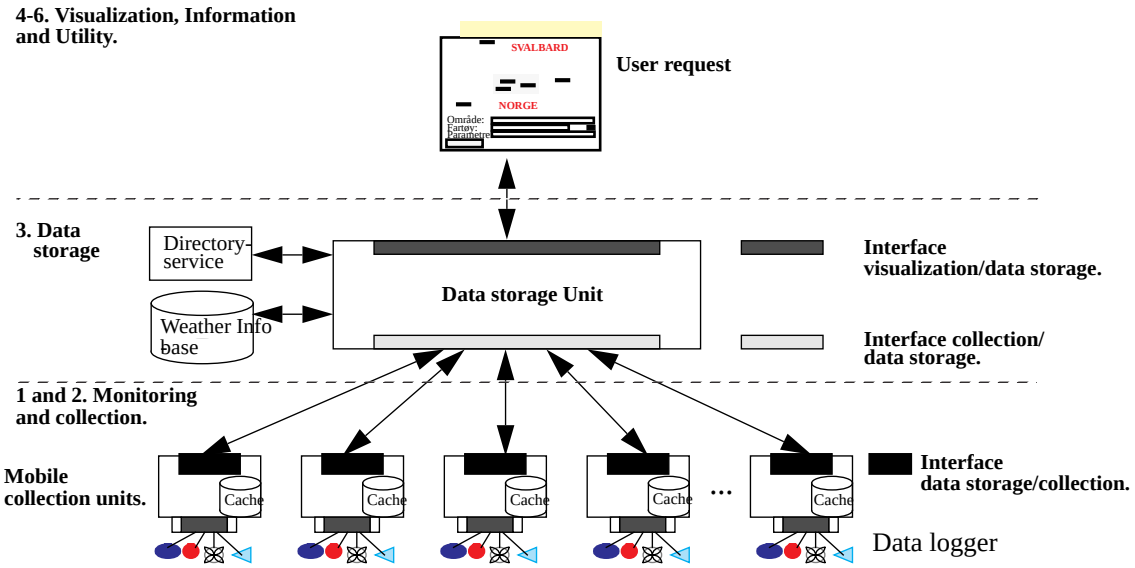


Figure 12. Application design

4.4 Implementation

The implementation is based on a heterogenous environment which consists of a Victor Personal Computer (80486) using MS Windows, a HP/9000 UNIX workstation and a DEC ULTRIX mainframe. The communication between data storage (ULTRIX mainframe) unit and mobile units (PC) are based on the developed mechanism. Communication between the visualization unit (HP/9000) and the data storage unit is based on TCP/IP.

4.4.1 The multicast and the multicast RPC - mechanism

The communication mechanism is, on the terrestrial side, implemented by a X-25 connection between the data storage unit and the LES. On the vessel side a satellite modem is required.

FFigure 13 on page 15 illustrates the implementation of the multicast protocol.

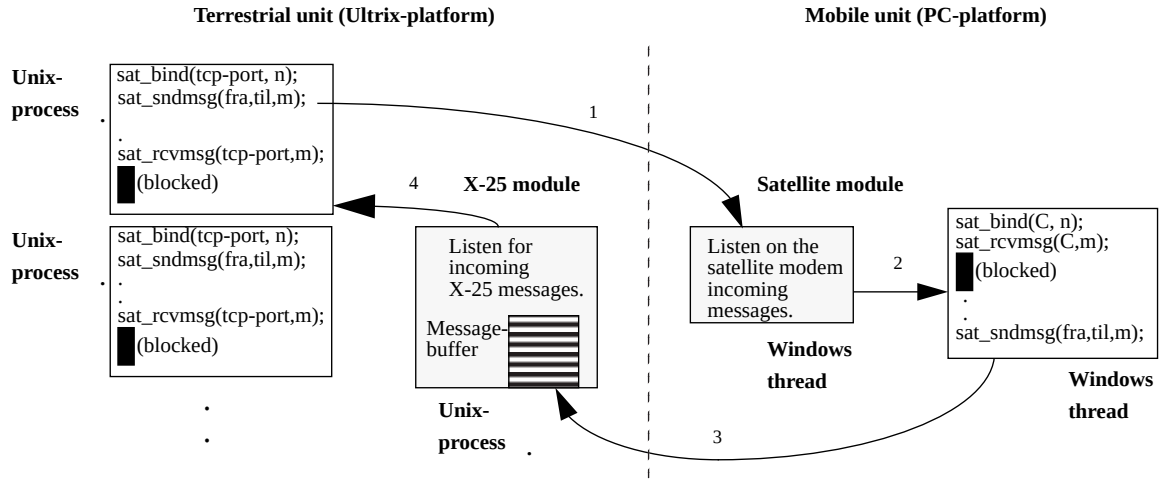


Figure 13. The Multicast protocol

In this figure we see how the multicast protocol primitives are implemented. The next figure illustrates the multicast RPC-mechanism. Each remote procedure are implemented as a stub

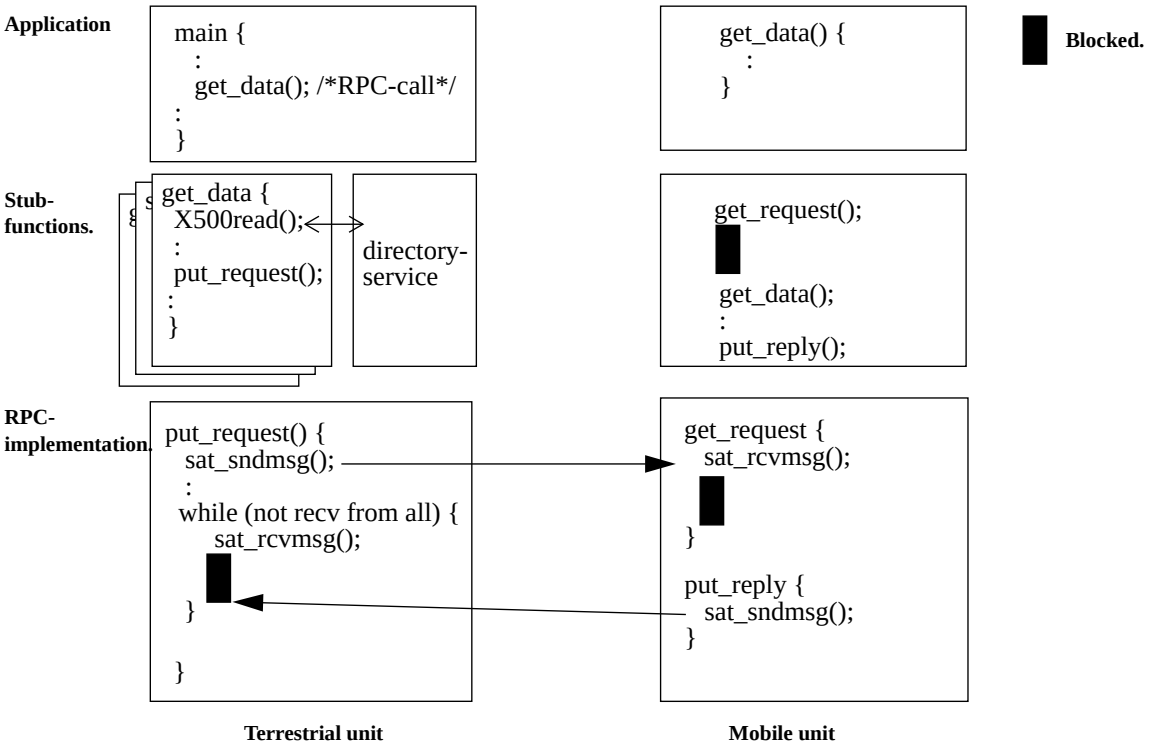


Figure 14. The RPC-mechanism implementation.

function on the client side. The stub uses the directory service to map names to subnet addresses. The communication mechanism is unsymmetrical because the data storage unit

has to handle many possible simultaneously connections while the collection units only has to handle one.

4.4.2 The application

The visualization unit are implemented as a Unix process and requests to the datastorage unit are based on a simple TCP/IP based RPC protocol. The data storage unit is a Unix process serving as a server for visualization units and a client against the collection units.

The collection unit is implemented as a MS Windows application that communicate with the satellite modem and perform the required actions. It checks the modem status in set intervals for incoming messages, and on user request pass messages to the modem and instruct it to send it to the given destination (data storage unit via the LES).

5 Test results

In this chapter the performance of the developed communication protocols is tested. These tests is not done to reveal errors, but to find out weather the protocols fills the requirements. There are several aspects that makes data transmission via satellite relatively slow:

- Low bit rate.
- The 270 ms delay to send ground-satellite-ground.
- The overhead of initializing a messaging channel.
- The overhead introduced by converting from DTE protocol to satellite link protocol.
- The land-earth-station must take turns to transmit.
- Retransmissions and message queues.

An overview of the test environment are illustrated in Figure 15..

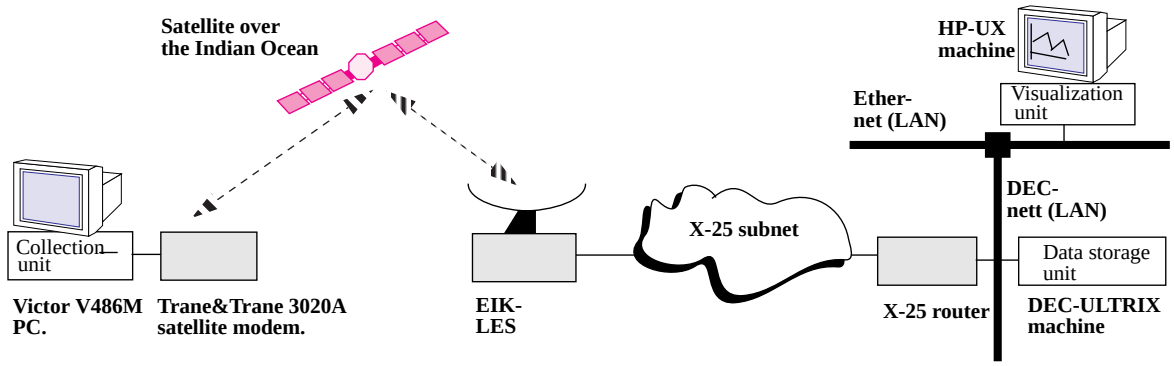


Figure 15. Overview of the test environment.

The most essential results here are the RPC test between the data storage unit and the collection units. This is presented in Table 1. The data storage unit uses the `get_data()` and `set_interval()` stub functions to communicate with the collection unit.

Figure 16. Test method

The results are averages of 50 measurements. Due to huge variations, median, max and min values are included. For comparison an estimated value are also included. The result are shown in Table 1. All values are in seconds.

	Message - Message					Poll - Message					Poll - Data report				
	Avg.	Med.	Max	Min	Est.	Avg.	Med.	Max	Min	Est.	Avg.	Med.	Max	Min	Est.
32 byte											84	92	97	37	1
100 byte	255	242	700	205	2,4	184	151	578	100	2,4					
1200 byte	269	250	802	210	18	198	165	584	130	18					

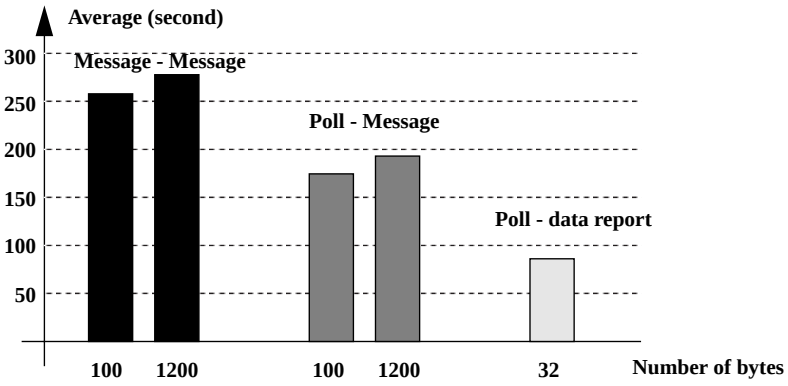


Table 1. Test result: message–message, poll–message, and poll–data report.

Estimated values are based on the formula:

$$\left(\frac{x_1}{9600} + \frac{y_1}{600} + 0.27\right) + \left(\frac{x_2}{9600} + \frac{y_2}{600} + 0.27\right)$$

where x_1 and x_2 are number of bits transmitted via X-25 (round trip). y_1 and y_2 are number of bits transmitted via satellite. The LAN delay is ignored.

6 Discussion and further development

The test results are based on averages of few repetitions (≤ 50). This due to practical limitations of the test environment. The test results showed that the LES and the satellite network are the limiting factor when it comes to performance.

Test results also showed that using polls for requests and data report as replies yield the best performance. The problem is that a data report is limited to 32 bytes. To utilize this the replies should be compressed. For bigger replies the message has to be sent as an ordinary text message.

If this system should be further developed it must be able to operate other LES and satellites (require a simple extension in the application). It should also be able to handle several groups (based on DNID's). More stub function calls has to be added to administrate a data logger. The visualization unit should be able to track a vessel in a given time slice. This could be used to predict a vessels movement in or out of coverage areas. The data storage unit could also be distributed.

7 Concluding remarks

The goal of this report has been to show that it is possible to integrate satellite communication (Inmarsat-C) in a distributed application based on the StormCast architecture, to yield greater availability to vital weather observations from the northern sea region and make a basis for further development of a application in real scale.

All these goals are accomplished. The Inmarsat-C system is an acceptable communication platform for this kind of application. The requirements for response time and reliable communication are limited.

The developed application will by all means cover the requirements that meteorologist today have. Observations are today available in limited amounts and frequencies (5-6 stations in the whole region), often several hours after it is measured. With this application weather observations are available in 2-3 minutes from a user specific area (or vessel). This will make weather prognoses more reliable and accurate. By using the Norwegian fishing-float / coast guard as a basis for collection of weather information an application could be implemented in full scale.

References

[ABBnera93] Specification of the ABB Nera Inmarsat-C Terrestrial Data Protocol. (REQSPEC3.WP), 10. mars 1993.

[Berg90] S.A. Berg, *"Inmarsat-C, Mobil satellittkommunikasjon via bærbare jordstasjoner"*, Teledirektoratet/Fagenhet for satellitt, May 1990. (In Norwegian)

[Birrel84] A.D. Birrell, B.J. Nelson, *"Implementing Remote Procedure Calls"*, ACM Transactions on Computer Systems 2(1): 39-59, February 1984.

[Enersen89] J. Enersen, *"Satellittkommunikasjon, metoder og muligheter"*, Svensk Sjøfarts Tidning, No. 40 1989. (In Swedish)

[Goscinski91] A. Goscinski, *"Distributed Operating Systems, The Logical Design"*, Addison-Wesley Publishing Company. 1991.

[Hanebrekke90] H.Hanebrekke, *"Datakommunikasjon til fiskefartøy"*, Fiskeriteknologisk forskningsinstitutt, ISBN-82-595-5914-5. Januar 1990. (In Norwegian)

[Hartvigsen88] G. Hartvigsen, D.Johansen. "StormCast – a Distributed Artificial Intelligence Application for Severe Storm Forecasting", In: M.G. Rodd, T.L. d'Epinay (Red.), *Distributed Computer Control Systems 1988*. Proceedings of the Eight IFAC Workshop (Vitznau, Switzerland, 13-15 September, 1988). Oxford: Pergamon Press, 1989. s. 99-102.

[Hartvigsen94] G. Hartvigsen, D.Johansen. "StormCast meteorological workbench", Department of Computer Science, Institute of mathematical and physical sciences, University of Tromsø, Norway, July 1994.

[Inmarsat93] Norwegian Telecom International Satellite Division. *"Inmarsat-C User Guide"*. June 1993

[Johansen91] D.Johansen, G.Hartvigsen, *"StormCast - a Distributed Application"*, In: Proceedings of the Autumn 1991 EurOpen Conference (Budapest, Hungary, 16-20 September, 1991). European Forum for Open Systems, Buntingford, Hertfordshire, U.K., pp. 273-286

[Johansen93a] D.Johansen, *"The StormCast Applied Approach to Distributed Computing - aspects on the Construction Phase of Prototypes of Large-Scale, Fault-Tolerant Distributed Monitoring Applications"*, Dr.Scient.-thesis, Department of Computer Science, Institute of mathematical and physical sciences, University of Tromsø, Norway. May 1993.

[Johansen93b] D.Johansen, *"The StormCast Applied Approach to Distributed Computing - aspects on the Construction Phase of Prototypes of Large-Scale, Fault-Tolerant Distributed Monitoring Applications"*. Addendum to Dr.Scient.-thesis. Department of Computer Science, Institute of mathematical and physical sciences, University of Tromsø, Norway. July 1993.

[Kistler91] J.J.Kistler, M.Satyanarayanan, *"Disconnected operation in the Coda File System"*, ACM Transactions on Computer Systems, 10(1):3-25, 1991.

[Patel90] A.Patel, V.Ryan, *"Introduction to names, addresses and routes in an OSI environment"*, Computer Communications, Vol 13, No 1, January/february 1990, s.23-33.

[Satyanarayanan90c] M.Satyanarayanan, E.H. Siegel, *"Parallel Communication in a large distributed environment"*, IEEE Transactions on Computers, Vol.39, No.3, March 1990, side 328-348.

[Shoch78] J.F. Schoch, *"Internetwork naming, addressing and routing"*, Proc. IEEE Computing, Conference IEEE, New York, USA, September 1978.

[Stevens90] W.R.Stevens, *"UNIX network programming"*, Prentice Hall, 1990.

[Tanenbaum90] A.Tanenbaum et.al, *"Experiences with the Amoeba Distributed Operating System"*, Comm. ACM, 33,12 (Des.1990), 46-63.

[T&T92] Trane & Trane, *"TT3020A, TT3020B and TT3022A Capsat Tranceiver for the Inmarsat-C Network"*, Software Interface Reference Manual.Version 1.4.Juni 1992.