"Y" A DISTRIBUTED RESOURCE SHARING SYSTEM

R. Popescu-Zeletin, K.-P. Eckert, V. Tschammer, W. Zimmer, L. Henckel

Hahn-Meitner-Institut für Kernforschung Berlin
Bereich Datenverarbeitung und Elektronik
Glienicker Str. 100
D-1000 Berlin 39

Abstract

The paper outlines the rationales for the development of the distributed resource sharing system Y at the Hahn-Meitner-Institute Berlin and describes its architecture.

The introduction of a distributed operating system is a prerequisite to a resource-sharing system. Y will not only provide the integration of networks of different qualities at hardware level (high-speed backbone, LANs of different technologies, ports to WANs: ISDN, X.25, satellites), but also provides the user an integrated global view of the whole system. It will be designed and implemented by decoupling the user's view from the hardware topology and characteristics by introducing a network wide distributed operating system and communication kernel.

I. ARCHITECTURE OF THE Y SYSTEM

I.1. BACKGROUND

The development of distributed processing has already a long history. The term "long" has to be evaluated from the perspective of computer history. If different theoretical ideas and paper works have witnessed these efforts for at least two decades, pilot systems have only emerged in the last years (TABS, ARGUS, GRAPEVINE,...).
This was motivated by the recent emergence of products and standards in network technology and data communication.

Our experience in network development lasts for 10 years from HMINET 1 to HMINET 2. The conclusion we may draw from practical implementations, operating networks in our campus, and an intensive work in the standardization bodies is that the present solutions and products are not satisfactory for the end user and the required qualities from a distributed system are only partially satisfied.
We learnt a lot in the last years, and the data communication community has improved its working tools for system structuring, among others by adopting a widely accepted reference model. Nevertheless, only very few pilot projects have reported on distributed applications and operating systems which are the prerequisite to resource sharing systems. Most networks and applications in operation only provide tools for bridging distances and partial heterogeneity and are not designed for resource sharing, better reliability and better performance.
Our environment is that of a typical research institute: several departments, heterogeneous interests, different data-flow paths internally and externally with a heterogeneous DP configuration interconnected by a medium-speed packet switching X.25 network and CSMA/CD networks. The network is ISO-OSI conform. The DP tasks span from real-time environment to word processing and from program development to theoretical computation. All these applications are served by programs with different data structures and different interfaces beeing attached to autonomous computer systems with different command languages, operating systems and hardware. The result of existing implementations is somehow schizophrenic: instead of helping the user through simplification, it has introduced more complexity. The user must not only become a DP specialist but a communication professional, too - even if he simply wants to get a certain computational result.

I.2. GOALS OF Y

The ultimate goal is to provide a unified global system for the end user where communication aspects, heterogeneity and topology are hidden. Y should ease the routine activities in the institute. This is complemented with support for real-time requirements for experiment control, better performance, reliability and availability in a heterogeneous DP environment.
Y aims to enhance the qualities of the actual DP environment by providing:

- homogeneous user interface to the different functions offered by the global system
- context-dependent functionality from the user point of view
- integration of different types of communication (voice, data etc.)
- flexibility and openess for new functions
- an architecture which allows a modular evolution of the system
- deterministic system behaviour for real-time applications
- high reliability and availability
- efficient and fair resource management

I.3 Y-ARCHITECTURE

To satisfy these requirements, the following methods and techniques will be employed in the development of the Y-system architecture.

The heterogeneity in hardware and system software, which is necessary for context related functionality and high performance, will be covered by standard operating system functions and data structures.
A unified control language will provide a homogeneous, abstract and context related user interface to the heterogeneous computing environment. High throughput and low response times shall be achieved by dedicated (front-end) processors and local area networks connected to a high-speed backbone facility. High-speed mass storage devices and database management systems will provide fast storage and retrieval of large amounts of data.
Gateways to WANs will allow access to external services, including public networks, ISDNs, satellites etc. External communications will be supported and controlled by dedicated management functions, performing user administration and service access control as well as providing security and protection.

I.3.1 User View

The top layer of the Y architecture provides an uniform user interface via a uniform control language and standard data structures.

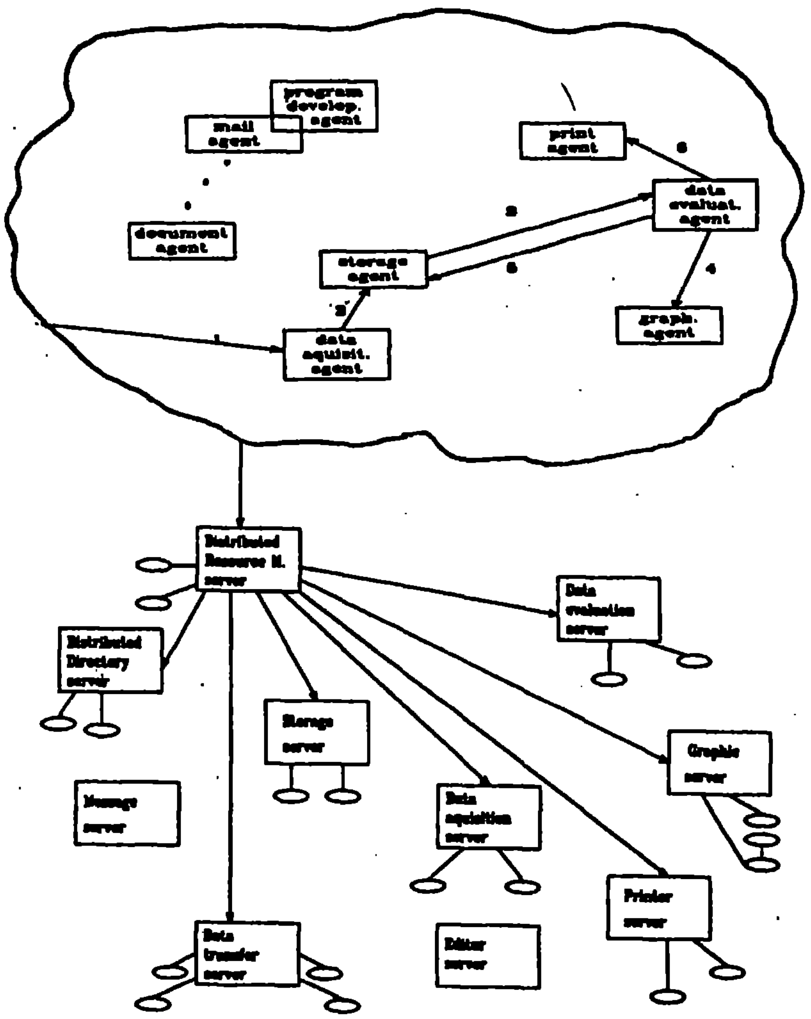

fig. 1: Agent / Server Configuration for a Special User Application

At this highest level of abstraction Y provides a pool of user agents representing the standard DP activities. A user application is modelled by configurating the required agents and their execution graph (sequence and parallelism of the agents operations). The agents match the different levels of service quality provided by the servers with the user's requests (fig.1). It is the user's responsibility to model his application by specifying the agents, their qualities and the execution graph.
The quality of service of an agent will be used in the underlying distributed operating system to identify and reserve those server incarnations which are able to provide the required quality of service.

The typed data structures for each agent are standard and self-describing in order to be able to transfer them from one agent to the other without user intervention. Note also that the user's view is independent of his location and the related system characteristics. Because the underlying levels of the system provide transparent communication it is not necessary that the entire functionality of the Y system resides in each computer.

I.3.2 Distributed Operating System View

The Distributed Operating System (DOPS) layer is the most important one in the Y architecture. It is the binding nucleus, where the two high technology worlds, computer and communication, are joined together to offer a new service quantity and quality (Multifunctional Computer Communication Aided Services).

To allow the resource sharing of all the installed components in a homogenous style the DOPS layer has to bridge the gap between the heterogeneous hardware and software characteristics of the products from the different worlds. The service offered by the Y system gives the user the illusion of a single integrated system. For the management and administration of the various resources, the DOPS layer is responsible for the overall control of the Y system. So as a refinement, the DOPS level can be seen from different points of view, taken by users, administrators or system developers. For every group it has to provide specific sets of essential and helpful tools for planing, organizing, controlling, maintaining or evolving their activities in an efficient manner.

Even though DOPS has to cover a broad spectrum of different requirements, it has to be efficient and reliable. From the user's point of view DOPS provides a uniform control language, which is adaptable to the problem context of typical user activities. The DOPS will decompose the user's applications (description of user operations on user data types) in DOPS operations and DOPS data types by a functional separation based on the current server environment. To guarantee the correct execution of the sequences of operations, as results of user operation decompositions, DOPS needs control over the servers and their incarnations. The physical locations of the server incarnations are not required from the users and are usually hidden to them. To locate a required service in the dynamically changing Y configuration, the mapping of name - address - route is provided by a distributed directory server (DDS). This leads for each user application to the related server configuration (fig.1).
An additional goal is an improved degree of reliability and performance. Each server in the DOPS layer is replicated and has several incarnations attached to different computer systems with different characteristics. The server distribution could be based on a "best-fit processing environment" strategy while the allocation of the server to the user agents should be based on "best-fit quality of service". Depending on available resources and their overall workload, DOPS has to match the user requirements with existing incarnations in the required quality. This task will be provided by the distributed resource management server (DRMS) in two phases. In the first phase the incarnations of the server configuration are reserved and bound to the user application. In the second phase the DOPS execution graph will be performed by the bounded server incarnations (fig.2).
Because Y represents a loosely coupled distributed system of autonomous, heterogeneous systems, DOPS has to cover a wide range of different policies. Components of the Y system may have their own area of responsibility or belong to the multi-administration responsibility of special DOPS controllers. There are several important research issues to be considered at the DOPS level:

- resource distribution and server allocation
- operations atomicity
- concurrency and recovery
- distribution of control
- security.

I.3.3 Communication Kernel

In order to provide the necessary communication support in the heterogeneous environment a uniform interprocess communication kernel must exist in each end system (fig. 2).

The requirements imposed on the communication kernel are more sophisticated as in usual interprocess communication since the Y communication kernel has to provide mechanisms for solving the heterogeneity of the different data representation, as well as different access, accounting and identification methods. A great help in providing all the characteristics above is the ISO/OSI Reference Model and its related standards. The communication kernel includes the entire ISO/OSI seven layers.

The decision of introducing standardized protocols for the communication kernel has two major advantages:

- the computer manufacturer support minimizes the maintenance effort of the entire system, and

- a connection-oriented communication kernel allows a better resource control at the DOPS level.

The initial set of standards supported by the communication kernel is:

- Global network layer
- ISO Transport class 0 (T70)
- Session BAS (T62), BSS
- Presentation/ Application (MHS, FTAM, 3X, ODA/ODIF)

I.3.4 Hardware

The Y architecture comprises a multifunctional network which integrates a variety of hardware components, ranging from very simple, dedicated end systems to complex, powerful multi-purpose facilities. It includes:

- Dedicated end-systems covering specific application areas such as program development, document processing and process control.

- High capacity, and complex devices and facilities for multifunctional services, like supercomputers or mass storage devices for number crunching, information retrieval, data bases, etc.

- Networks and gateways which interconnect these components and provide access to external services. Specific investigations, developments and installations are necessary to realize these components:

Local Area Networks. Functional tests and performance investigations which evaluate the capabilities of existing networks regarding the transfer of voice, video and digital data under the specific requirements of the different application areas. Development of new, improved techniques which particularly satisfy the requirements of real-time applications.

Backbone Facility. Evaluation of products and techniques which propose a theoretically unlimited throughput via a large number of parallel channels in order to satisfy the requirements for a high-speed backbone facility which integrates the different components and communication services.

Gateways. Development of a modular, intelligent gateway which serves as an interface between end systems and the backbone facility as well as a relay between interconnected subnets.

The resulting architecture of the Y-system is illustrated in figure 2.

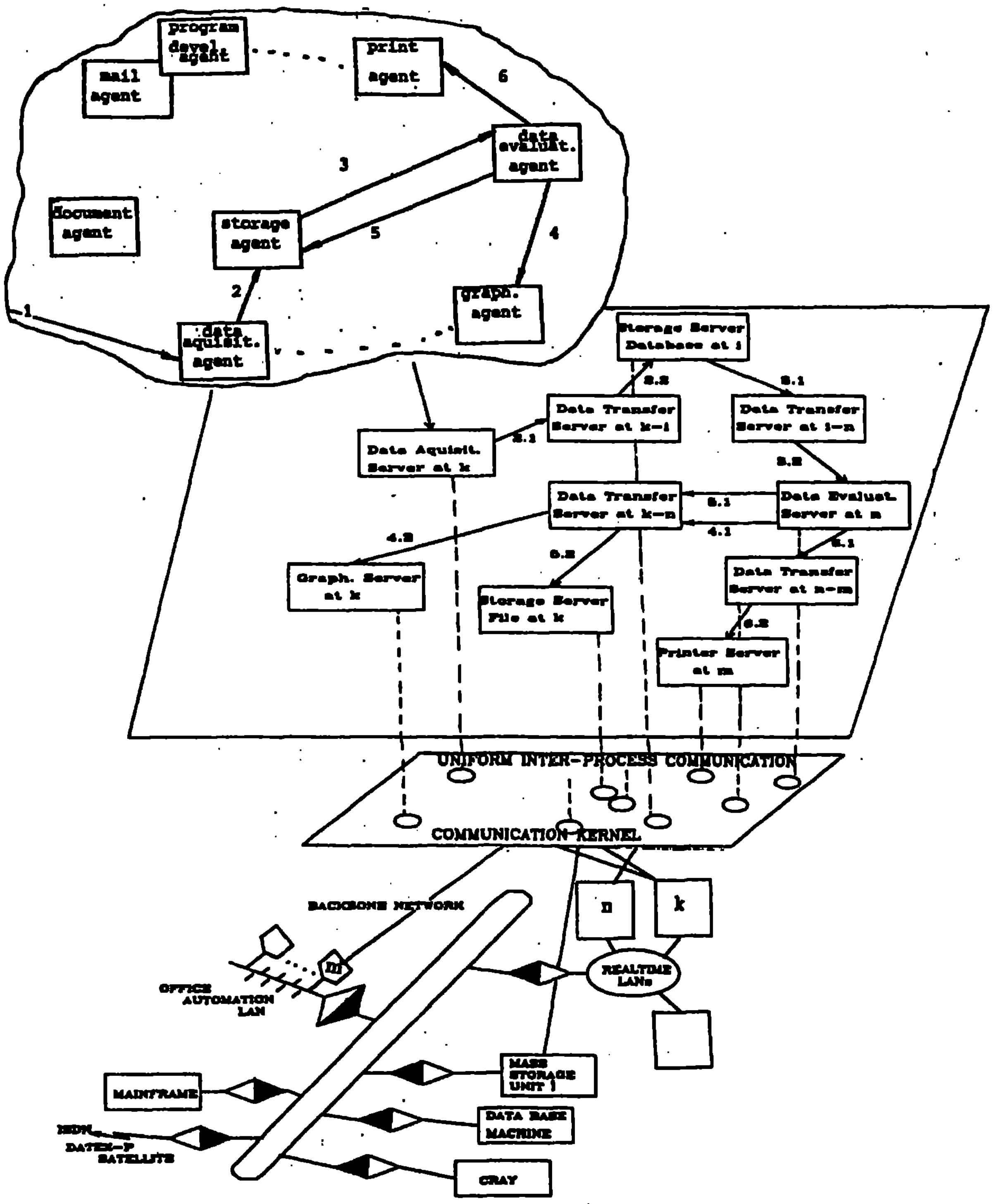

fig. 2: Architecture of the Y System