PROCEEDINGS OF SCIENCE

The CWB Network Information exchange Environment (NICE)

Mark Cheng[1]
Central weather bureau
64 Gongyuan Road, Taipei, Taiwan 10048,Republic of China
E-mail: mark.cpcheng@gmail.com

After years of development of operational meteorological data processing systems in CWB, dozens of automated application systems, with different data processing contents yet similar operational behaviors, were developed. To support the development of automation mechanisms and reduce the maintenance efforts of these different meteorological data processing systems, A network based application framework for real time distributed data processing, the Network Information exChange Environment(NICE), was developed to ease the job and strengthen the system operations. The NICE consists of nine components. (1) Information Transport facility - IT, (2) Process Automation facility - PA, (3) Data access and Management facility - DA/DM, (4) Information Service facility - IS, (5) MONitoring and LOGging facility - MON/LOG, (6) Fail Over facility - FO, (7) User Interfacing facility - UI, (8) Hierarchical Storage Management system interfacing facility - HSM, (9) HPC interfacing facility - HPC. These nine components integrate the past experiences and best practices on operational mechanism development in CWB into an unified application framework and toolkits for optimal application system development and operations. The NICE utilizing open system software technologies, packages and standards into the building blocks of NICE components to speed up application development while maintaining the application system's flexibility, operability, reliability and interoperability among systems across network.

The International Symposium on Grids and Clouds and the Open Grid Forum
Academia Sinica, Taipei, Taiwan
March 19 - 25, 2011

[1] Speaker

PoS(ISGC 2011 & OGF 31)082

1. Background

The Central Weather Bureau started her modernization projects in 1984 to promote the numerical weather prediction (NWP) abilities of Taiwan. In the project, the first high performance computer system was introduced to CWB as well as a series of meteorological data processing and NWP systems were developed on specific proprietary computer systems. Start from 1990, the UNIX based open systems were introduced to CWB, networking and clustering of computers to resolve the application needs became a basic approach. All the existing software application systems were ported to the open system environment and many new data processing systems were continuously developed. Fig.1 and Fig.2 shows the network environment and the data flow of major data processing systems in CWB. After few more years of development, dozens of automated application systems were developed and operated in the network based open system environment. Though many of the operational behaviors were similar for those many different systems, nevertheless, the implementation of operating mechanisms were very different which introduced another level of difficulties to operate and maintain those many systems. About the same period, the distributed computing concepts and environments were introduced and evolved [1][5][6][7]. The interoperability [2] and comply with standards [3][4] were emphasized. Borrow many of the DCE architecture and operation concepts, a network based application framework for real time distributed data processing, the Network Information exChange Environment (NICE), was developed to reduce the system operating mechanisms development efforts and to easy the operation and maintenance support of those many different meteorological data processing and application systems in CWB.

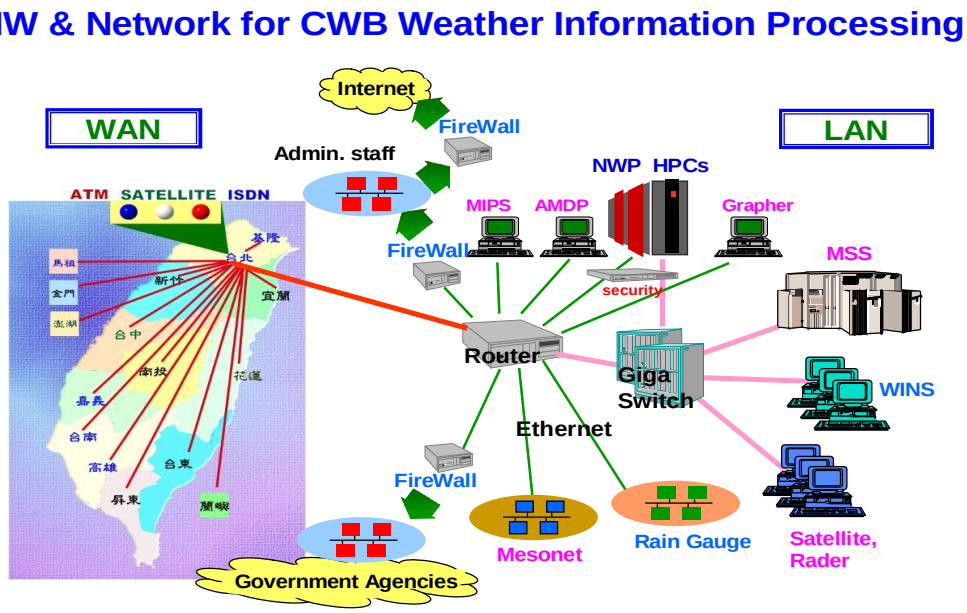

Fig.1. The hardware and network environment for CWB weather information processing

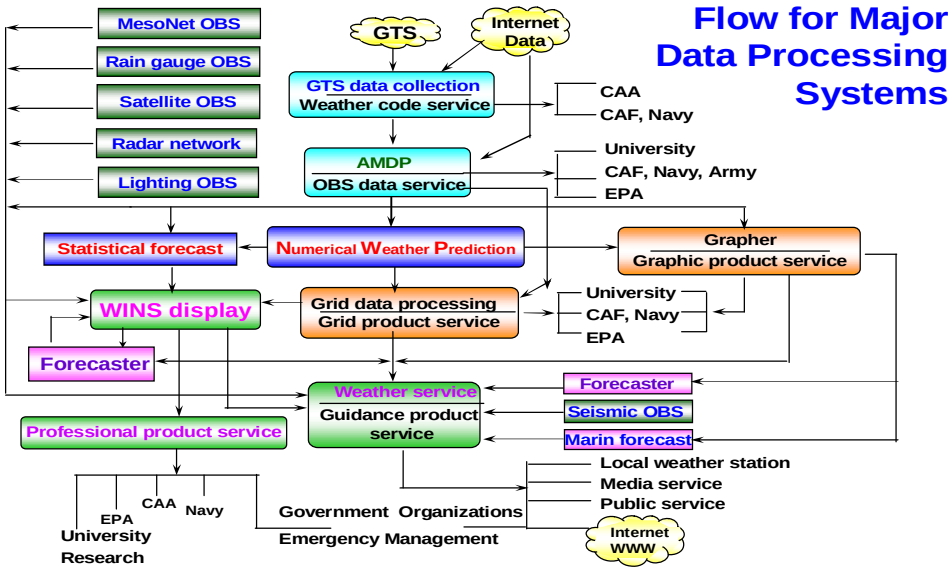

Fig.2. Data flow for major meteorological data processing systems in CWB

2. The NICE architecture and components

The NICE consists of nine components, as shown in Fig.3, which integrate the past experiences and best practices on operational mechanism development in CWB into an unified application framework and toolkits for optimal application system development and operations. The NICE utilizing open system software technologies, packages and standards into the building blocks of NICE components to speed up application development while maintaining the application system's flexibility, operability, reliability and interoperability among systems across network. A short description to each components are followed：

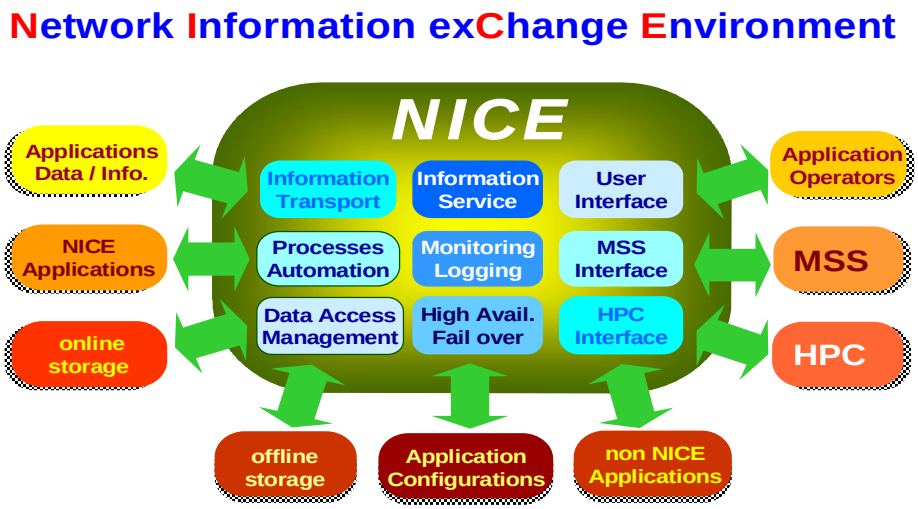

Fig.3. NICE components

(1) Information Transport facility - IT

IT rovides the solutions for reliably exchange of data and messages in different hosts within or among systems across network. Access controls can be applied to information objects and hosts if so wished. Major IT components include : (a)resource broker, (b)connection aids, (c)message transport, (d)Information routing, (e)remote device access, (f)binary data conversion.

(2) Process Automation facility - PA

PA provides the solutions for automated processes management and processes flow control. The process activation mechanisms include : (a)time driven (one time or periodical), (b)event driven (single event or logical combination of multiple events). The process management functions include : (a)creation/deletion/rerun, (b)processes queueing, (c)number of processes, (c)execution elapse time.

(3) Data access and Management facility - DA/DM

DA/DM provides the solutions for on-line data records and files access in local or remote hosts across network. Files can be either plane or indexed. File management mechanism includes : creation, purging, archiving, backup, restore.

(4) Information Service facility - IS

IS provides the solutions for reliably receiving and sending information objects with standard FTP, RCP and LDM protocols to interface the outside (non-NICE) world. Sending/receiving status detection, Timeout, retry, bandwidth control, multithreading and queueing mechanisms are provided to support the reliable transport of information objects.

(5) MONitoring & LOGging facility - MON/LOG

MON/LOG provides the solutions for detecting, monitoring, logging and reporting resource utilization and/or processes activities events of application systems. The events include : (a)resource usage & utilization, (b)data availability, (c)network connectivity, (d)process life cycle, (e)activities timing. Different severity level could be associated to events which can be centralized managed, displayed and reacted with an operator UI.

(6) Fail Over facility - FO

FO provides the solutions for maintaining high availability of system operation. Critical resources and services can be configured, monitored, managed and failovered by a state machine based decision making kernel with the support of activity monitoring, abnormal detection and fault isolation.

(7) User Interfacing facility - UI

UI provides the solutions for application users and operators to interface with the system. A set of X-window based widgets include : (a)menu items, (b)selection list, (b)command button, (c)file selection, (d)text entry, (e)dialog box, (f)message window can be configured and composed via a scripting language for system's specific interfacing needs.

(8) Hierarchical Storage Management system interfacing facility - HSM

HSM provides the solutions for interfacing with the hierarchical storage management system, a combination of near-line tape library and cache storage server with HSM functions support, in CWB. Inventory query to storage pool, migration/de-migration control and basic

file operations can be applied to file objects in local and/or remote storage devices and/or servers.

(9) HPC interfacing facility - HPC

HPC provides the client side solution for interfacing with the High Performance Computer system in CWB. A Simplified commands set to encapsulate different batch systems commands for most commonly used commands for Job submission, execution control & status query.

In addition to the above components, The NICE also provides toolkits for the installation of NICE packages, program specification documents and on-line manuals generation as well as date/time calculation, environment variables & string manipulation and many other utilities & API's to increase the development productivity.

3. NICE application framework and operation concepts

Fig.4 shows the application framework and operation concepts of a typical application system which utilizing NICE components. The elliptic shaped icons, in green, represent the components provided by NICE framework. The square shaped icons, in blue, represent various kinds of data processes provided by the application system. The drum shaped icons, in yellow, represent message and/or data storages in the network to be accessed by the processes of the application system. NICE protocols network is basically a TCP/IP network with standard TCP/IP service protocols and some specific service protocols defined by NICE. The core component of IT is the message router(IR) which sends and receives all kinds of information objects(data packets) need to be processed by some specific application systems. The message processor is a set of user defined processes attached to message router to process various information objects. The data processor is another set of user defined processes, which could be managed by the process manager(the core of PA), to transform the data into products. The data processor could also utilize the DA facility to access the distributed data and/or product stores somewhere in the network. The file manager maintains the space balance required for the data files and storage spaces in the system. The activities monitor, a component of MON/LOG, detects and logs the resource usage and process status of the system. The user defined channel processors utilize the connectivity tools of IT to talk to the processes of some other systems in the network to perform the required operations. The user defined service processors prepare the necessary conversions and/or preprocessings for those products which need to be delivered using standard protocols to the other system by the information service process of IS. The user defined protocol converters are processes which may be required for some system to bridge the NICE and non-NICE protocols. Configuration / logging / monitoring / operation interfaces and high availability fail over control are the components to support the over all system operations and robustness.

The development efforts for the application system developers, who are going to utilize NICE components and/or framework for his(her) system development, are to prepare the specific processors for the specific system and to select and configure the required NICE components according to the behavior and needs of the specific system.

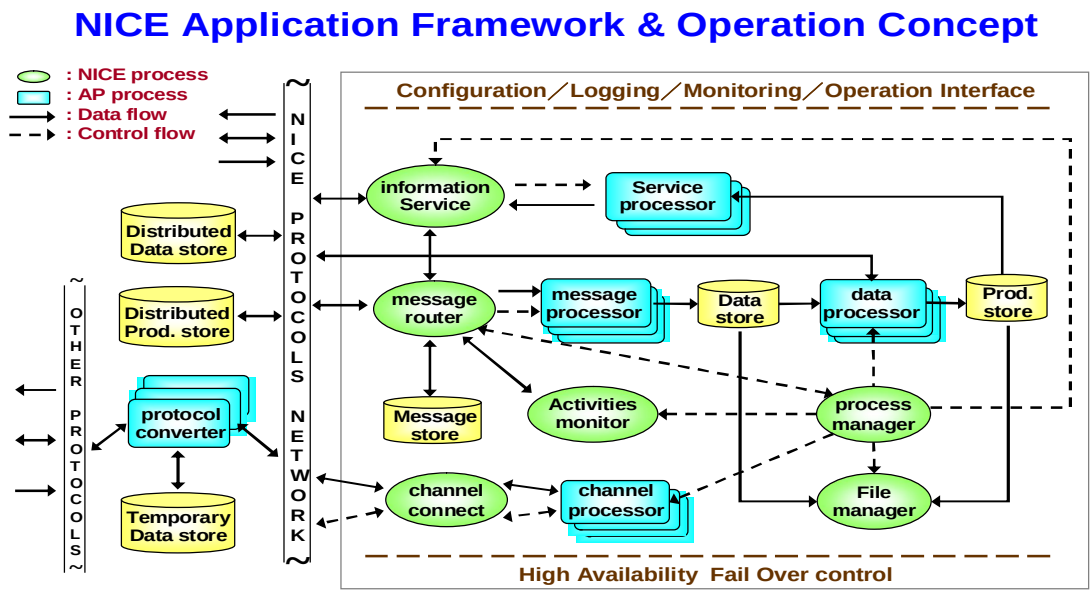

Fig.4. NICE application framework and operation concept

As shown in Fig.5, the NICE components can also be mapped to the famous gird architecture defined by Ian Forster etc. [8] as well. A NICE Application, which utilized some part of NICE components and span on single or multiple hosts, could make use of suitable IT(IR/Nr/Ns/DT/Ndq/Dio/HS) components to communicate to different processes in different hosts while uses components in IT(Nlap)/DA/MON/MSS/HPC to access resources with in the same application system or resources on the network as well as uses PA to automate the whole processes flow. Different NICE applications can also collaborate to each other using NICE components to exchange messages and/or data to meet the meteotrological data processing needs of CWB.

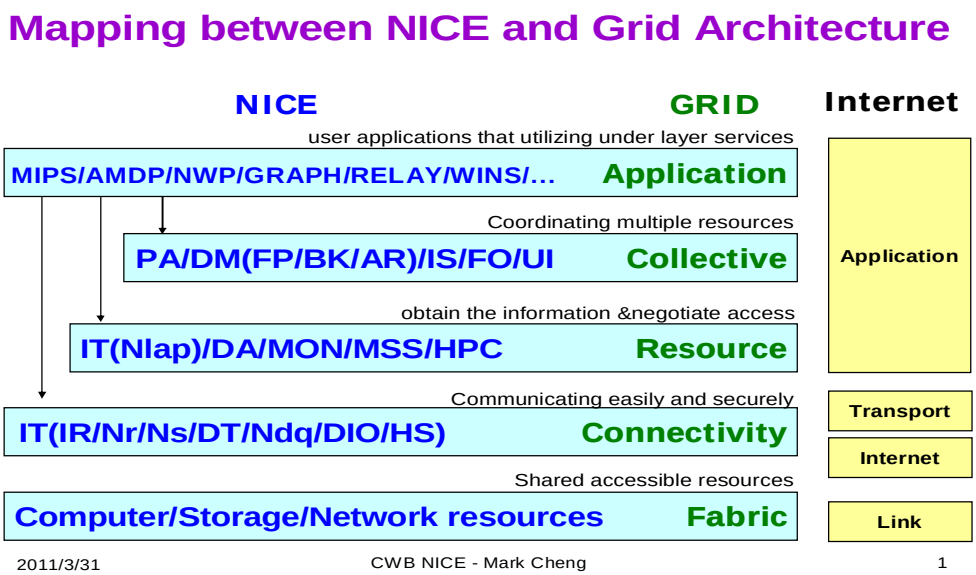

Fig.5. Mapping between NICE components and grid architecture

4. Real world application and future

The NICE has been exist, utilized and continuously improved for about ten years in CWB. Currently, 147 hosts of 39 systems were installed and running some components of NICE. Every day, over 1.2 million information objects are routed by NICE message routers and over 150 thousand processes are managed by NICE process managers. The storage volume managed by NICE file manager around 20TB and the data access transaction per day over 6 million records(files) while the transaction volume well exceed 10TB. The product files delivered by information service to end users and/or down stream systems outside CWB are over 35 thousand files and exceed 160GB.

Still, the NICE needs to be enhanced to increase it's support to web browser based UI and Linux HA based fail over control. The NICE also needs to implement the support for the new HPC to be procured in the next few years. The distributed computing concepts and toolkits are advancing continuously [9][10]. New technologies and concepts could be incorporated into NICE. In the future as well.

5. Acknowledgement

NICE was first designed and developed in the CWB phase three modernization project (1995.7~2001.6). Continuous maintenance and enhancement followed, nevertheless all the associated technical documents were written in Chinese. A group of skillful application system developers were chosen to implement the NICE framework. With out their dedication and contribution, the NICE would still be in the air and the application systems may still work as isolated islands in CWB network.

References

[1] Brad Curtis Johnson：A Distributed Computing Environment Framework: An OSF Perspective. *DEV-DCE-TP6-1*. June 1991

[2] Open Software Foundation：Interoperability: A Key Criterion for Open Systems. *O-DCE-WP-18-1*, November 1991

[3] Open Software Foundation：The OSF Distributed Computing Environment: Building on International Standards. *O-DCE-WP-19-1,* April 1992

[4] H. Lowe, J. Quigley：Mapping DCE to Minimal OSI (mOSI). *OSF DCE SIG RFC 43.0*, July 1993

[5] R. Salz：DCE 1.2 Contents Overview. *OSF-RFC 63.3*, October 1996

[6] Ward Rodenberry：DCE today: An indispensable guide to DCE. 650pp, The Open Group, 1998.

[7] Kendall, S.C., Waldo, J., Wollrath, A., and Wyant, G.：A note on distributed computing. Sun Microsystems, Technical Report TR-94-29, 1994

[8] Ian Forster, Carl Kesselman, Steven Tuecke：The Anatomy of the Grid – Enabling Scalable Virtual Organizations. *Intl J. Supercomputer Applications*, 2001.

[9] Li, Maozhen, Mark A. Baker：The Grid: Core Technologies. Wiley, 2005

[10] Heinz Stockinger：Defining the grid: a snapshot on the current view. *J Supercomput*, 2007