MVS/ESA coupled-systems considerations

by M. D. Swanson
C. P. Vignola

One of the most important future MVS environments will be to provide on-line transaction processing and decision support through the use of large data "warehouses" in heterogeneous networks. To provide the large capacity and high availability required for this environment, it will be necessary to use multiple, cooperating MVS systems to provide those services. The design of such coupled MVS systems must accommodate a number of factors related to both the general environment and the user's needs, including availability, growth, granularity and scale, systems management, migration and coexistence, and single-system images. In this paper, we describe the system services available with MVS/ESA™ SP Version 4 that can support the coupled-systems environment, and discuss how those services can be exploited.

Introduction

As customers increase the amount of business which they entrust to computer systems and applications, the importance of the availability of those systems and applications increases dramatically. In particular, as the computer user community changes from data processing professionals, technicians, and operators to business professionals and corporate customers, applications are required to be continuously available. To achieve this, both unplanned and planned outages must be addressed. Availability improvements in single-system hardware and software products will continue to be made. However, there will still exist a class of failures which will result in single-system outages. Many planned changes can be performed on a running system, but changes to the basic, central parts of single systems, such as those required to perform a major hardware or software upgrade, require taking those systems down. For both of these reasons, to achieve continuous availability, a data processing configuration comprising multiple cooperating systems is required. Furthermore, it is required that while operating in close cooperation (i.e., as coupled systems), these systems do not have common points of failure.

While the power of the largest available single system has been increasing at an impressive rate, the workloads of some customers are still growing beyond the scope of any single system. Customers have traditionally managed this growth by partitioning workloads. This almost always requires significant planning and systems management, and sometimes requires modification of the application. The effort customers spend on these activities detracts from the business goals of the customer and often impedes or delays further application development. The ability to leverage additional systems against workload in a manageable fashion provides a much more productive alternative.

In addition to growth considerations for overall capacity, the MVS design solution must address requirements of scale and granularity. Implementation of the coupling functions must be feasible over the entire range of system sizes from smallest to largest. Furthermore, support must be provided which enables customers to grow in increments which meet their business needs. To address the scale and granularity requirements, support must be provided for coupled configurations comprising collections of small systems, collections of large systems, and collections of systems which are a mixture of small and large systems.

Traditionally, configurations comprising multiple MVS operating system images have been significantly more complex for customers to manage. Reducing the complexity and the associated cost of managing multiple MVS systems which are operating in concert will require fundamental changes in operating system services which provide data access and work scheduling services. Additionally, complexity must be reduced in the areas of customization, operations, performance, and availability management.

Changing from existing system configurations to an initial coupled-systems environment, and subsequently upgrading as new coupling functions become available, will require coexistence of multiple levels of hardware and software as well as nondisruptive migration support. Support is required which enables at least two levels of hardware and two levels of software to be intermixed within the coupled-systems environment. New functions enabled by the higher level of support may be made operational only on the upgraded systems.

Over time and where desirable, the collection of MVS systems coupled together to provide a single configuration will be made to appear as a single data processing capability. From the end-user point of view, the coupled systems will be seen as a single server providing access to data, and to functions which access and manipulate that data. Systems management activities concerned with operations, performance measurement and tuning, and work distribution will have a single image. Some areas of configuration management and customization may also present a single image across the coupled systems. Support will continue to be provided where required to address specific hardware and software elements of the configuration. For example, support will be provided for reconfiguration and service activities which are related to a specific element.

The MVS coupled-systems configuration will be a provider of data access and program services within the overall wide-area network. Communications outside the coupled-systems environment will be provided by networking products supporting a wide variety of protocols. MVS services are regarded as one element within the overall distributed environment, but considerations for network and distributed processing functions outside that element are beyond the scope of this paper.

The first stage of MVS coupled-systems support is provided by MVS/ESA™ SP Version 4 [1]. In this release, basic system services required for a coupled-systems configuration are made available. In the following section, this basic operating system support for coupled systems is described. In a subsequent section, an overview is given on how those services can be exploited. For each area which exploits one or more base system services, a description of the functions made available by that area is also given.

Base system services

MVS support for the coupled-systems environment is provided as part of the basic operating system. These system services exist in all occurrences of MVS, starting with MVS/ESA SP Version 4. The collection of systems which are coupled together through the base operating system functions is termed a *systems complex*, or *sysplex*. The functions described below provide an interface which is independent of the underlying System/390® architecture and hardware implementation on which they are initially built. It is intended that the interfaces and functions provided will be maintained over time and will take advantage of future architectures and new hardware implementations.

In the coupled-systems environment, many services will be provided by independently executing programs running on multiple occurrences of MVS. These independently executing programs will cooperate to provide a single type of service. There are several models that these independent programs may exhibit—*peer-to-peer*, *active-backup*, *functional specialization*, and *client-server*. In the peer-to-peer model, a given function may be replicated on each MVS system within a sysplex. This type of structure is exemplified by the global resource serialization component of MVS. In the active-backup model, the backup becomes the active provider of service in case of a failure of the active. (An IMS environment with XRF is one such configuration.) In the functional specialization model, the different functions of a service are distributed on the systems in the sysplex. JES3 Global and Local functions and CICS™ TOR (terminal-owning region) and AOR (application-owning region) functions are two examples of such configurations. In the client-server model, one program provides the application programming interface (API) of the service, while the other program provides the function of the server.

The collection of program instances which provide a given service is called a *group*. An individual occurrence of a program function within a group is called a *member*. Each group within a sysplex has a unique name. Each member in a group has a name which is unique within the group. Many of the same basic services required by cooperating programs executing on a single system are required by members of a group which may span multiple MVS systems. Among these services are the ability to know when members become active and stop being active in the group, the ability to know when members of the group terminate abnormally, the ability to know the status of members of the group, the ability to pass messages among members, and the ability to access synchronized time. Base MVS coupling services provide these capabilities.

- *Group and member services*

The process of becoming associated with a group is supported by two MVS services. The *create* service provides a programming interface through which a member of a group can be defined. As part of this definition, the user-defined state associated with the member may be initialized. (More information about the user-defined state can be found in this section, following Table 1).

The *join* service enables a program to utilize the functions provided by MVS coupled-systems support. A group name and optional member name are input to the *create* and *join* services. Following successful invocation of the *join* service, monitoring and signaling services are available to the program. (See the sections on monitoring services and signaling services for additional information.)

An option provided on the *join* service enables permanent recording of the status of a member. Information regarding the last known state of the member is returned from the *join* service. This state information can be used to determine how the member terminated during its last execution. If it is known that a program terminated normally, recovery processing when it is restarted may be eliminated or greatly simplified. The MVS address space and task which is active at the time the *join* service is invoked are used to determine the state of the member.

A program may discontinue the association with MVS coupling services in one of three ways. First, normal termination of the member may be initiated through the *leave* service. On completion of the *leave* service, MVS coupling services can no longer be used by the program, and no status data regarding the member will be maintained. Second, a program may terminate normally and have status information maintained by invoking the *quiesce* service. The *quiesce* service may be used only by programs which have requested permanent status recording. Finally, a member is determined to have terminated abnormally if termination of the task, address space, or system associated with a member occurs before the *leave* or *quiesce* service is invoked by the program. Failure of a member with no permanent status recording results in that member becoming not defined to MVS coupling services. Failure of a member with permanent status recording results in that member being placed in a failed state.

Table 1 Member state flow.

Current state	*Action*	*Resultant state*
Not defined	Create	Created
	Join	Active
Created	Delete	Not defined
	Join	Active
Active (permanent recording requested)	Did not leave group at termination	Failed
	Quiesce	Quiesced
	Leave	Not defined
Active (permanent recording not requested)	Did not leave group at termination	Not defined
	Leave	Not defined
Failed or quiesced	Join (restart)	Active
	Delete	Not defined

Status data for a member which is not actively utilizing MVS coupling services may be removed from MVS coupling services through the *delete* programming interface. The status data could have resulted from use of the *create* service with no further action, use of the *quiesce* service, or failure of a member which had requested permanent status recording.

In **Table 1**, the five member states recognized by group and member services are identified. For each state, the action which causes a state transition is shown along with the resulting state. Notification of changes in state are delivered to all interested members within the affected group.

While the above states are generally useful, they may not adequately describe all the conditions supported by a program. Therefore, associated with each member, MVS coupling services supports a user-defined state. The user state can represent any condition supported by the member and has no meaning to MVS coupling services. MVS coupling support provides services which enable a program to change the user state associated with any member which is in the same group as the requesting program. The *set user state* service optionally accepts a value used to match against the current user state before a new user state value is set. Notification of changes to a user state value are delivered to all interested members within the affected group.

Notification of changes in the state of members within a group may be provided to all members within the affected group. A program indicates that it is to receive notification of state changes by providing a *group exit* to the *join* service. Events presented to the *group exit* include changes in the MVS group services state of a member in the group, changes in the user state of a member in the group, and changes in the state of systems within the sysplex.

Delivery of notification events is controlled through MVS coupling services. The *group exit* is scheduled as a local service request block (SRB) in the primary address space that is current at the time the *join* service is invoked. Execution of the *group exit* is serialized by MVS coupling services so that only one occurrence of the exit is in control at any point in time. Events are presented to the exit one at a time through a parameter list provided to the exit interface. MVS coupling-services modules running under the SRB receive control before and after the exit code. When the exit code has completed processing of an event, control is returned to MVS, which then determines whether another event is awaiting presentation. If no events are awaiting presentation, the SRB terminates. If events are awaiting presentation, the exit code is given control with an updated parameter list, and SRB execution continues.

While it is required that events be presented in a logical order, it is not required that all events be presented. Therefore, a filtering of notification events occurs in MVS to ensure logical ordering. Logical ordering is enforced for changes in MVS group services state, changes in user state, and changes in system state. For example, two changes may occur to the user state of a particular member. If both of those events are reported, they are reported in the order of their occurrence. If only one event is reported, it is the later of the two changes in the user state. Similarly, a *join* event, if reported, is always reported before a *leave* event. Events which report the termination of a member—*leave*, *quiesce*, *delete*, and member failure—are always reported.

Information regarding the status of groups, members, and systems within the sysplex may be obtained through the *query* service. This information is available to any authorized program. System-level information includes a list of the systems that currently make up the sysplex, and the system state of each system (e.g., operational, being removed from the sysplex, not operational but not yet removed from the sysplex). Information regarding existing groups and associated members may also be obtained. For a group, identification of the set of associated members is provided. For a specific member, information includes the MVS group services member state and user state.

MVS coupling services provides an interface through which any member of a group may initiate the termination of any other member in the same group. The function requested of the *terminate member* service does not complete before control is returned to the caller. As part of this processing, a signal may be sent to the system on which the target member is executing. The MVS system on which the target member is executing initiates an abnormal termination of the task associated with the target member. Completion of the *terminate member* service is reported through member state change notification.

- *Monitoring services*

The ability to recognize changes in the operational state of elements contained in a sysplex is a fundamental requirement in order to initiate recovery actions for failed elements and to support and take advantage of added elements. Within a single system, the ability exists to detect the creation and termination of tasks and address spaces. The monitoring services provided by MVS coupled-systems support provides the ability to detect changes in the operational state of systems within a sysplex and of members within groups.

Operation of the system monitoring functions of MVS coupling services is controlled through two parameters. The first parameter defines the amount of time that may elapse between events which mark the successful execution of a system and is termed the *failure detection interval*. The second parameter defines the amount of time that may elapse between the occurrence of an event which marks the successful execution of a system and operator notification of that event. This parameter is termed the *operator notification interval*. Both the failure detection interval and the operator notification interval are initially specified at system initialization time. During ongoing system operation, either of these values may be changed by operator command.

These installation-specified intervals are used to determine the actions taken during execution of the system monitor. The system monitor is given control as the result of an external timer interrupt established for a one-second interval. Each time the system monitor is given control, it performs processing unique to the system on which it is executing. Every third invocation, the system monitor makes ready a waiting task to make the system status visible to other systems in the sysplex.

The system monitor task causes system status to be maintained in the coupling data set. (See the section on system services characteristics for a discussion of the coupling data set.) Processing in each system obtains a time stamp, causes that time stamp to be written in the coupling data set, and causes the time stamp of all other systems to be retrieved from the coupling data set. Each system then determines what actions should be taken.

- If new systems initialize within the sysplex, a notification event is formulated for delivery to all members which have provided a *group exit*.

- If a system has not updated its status within the installation-defined failure-detection interval for that system, ownership of the detected error for that system is established. Ownership for a detected error is achieved by updating the recovery record on the coupling data set. The recovery record is a single record which contains an entry for each system reflecting ownership of errors detected for that system. Updates to the recovery record are serialized through MVS coupling services which access the coupling data set. Each system in the sysplex executes the system monitor algorithm. Ownership of a detected error is established by the first system to successfully update the recovery record identifying itself in the recovery record entry belonging to the system for which the error has been detected.

  When a system does not update its status within the defined failure-detection interval, a system-status-update-missing notification event is formulated for each member on the system which is not operational. System-status-update-missing notification for a member is delivered to all members which are in the same group as the member on the system experiencing the error.
- If a system which has been detected as having a system-status-update-missing event continues not to update its status at the expiration of the operator notification interval, an operator message is issued to determine the status of that system. If the operator responds that the system has failed and been reset, a notification event is formulated to report the termination of that system and all members on that system. The operator may reply that the system is temporarily not operational for some duration of time. Monitoring continues for a system that is temporarily not operational. If the system has not updated its status by the end of the operator-specified interval, the operator-notification sequence of events is repeated.
- If a system resumes updates to its system status after having been detected as having a system-status-update-missing event, a system-status-resumed notification event is formulated for each member on the system which has resumed operation. System-status-resumed notification for a member is delivered to all members which are in the same group as the member on the system which has resumed execution. If an operator message had been issued, it is deleted.

A system that has established ownership of a status-update-missing condition may become not operational before the condition is removed by either an update of the status of that system or an operator reset. Ownership of the error for the system which becomes not operational is assigned as described above. Additionally, this condition causes some other system within the sysplex to assume ownership of the detected errors being processed by the system for which the not operational condition has been detected.

While the system monitor effectively detects when a system is not providing operational services as requested, cases may arise under which a system is operational but occurrences of a program executing on that system are not providing acceptable service. Monitoring services for group members are provided to address this class of failures. When the *join* service is invoked, three values may be provided which will enable member monitoring. One value defines an acceptable time interval between events which indicate the successful execution of the member. A second value provides the address of an area in MVS common area storage which is to be changed within the specified interval to indicate the successful execution of the member. The third value specifies the address of a *status exit* which is to be given control should changes to the specified common area not occur within the interval defined.

Member monitoring occurs as an extension of the system monitor. During local system processing for the system monitor, each member having requested monitoring is processed. Processing begins by accessing the common area storage location. If the value has changed from the value obtained when the common storage area was last accessed, the member is considered to be operating successfully. If the common area remains unchanged across the expiration of the member monitoring interval, the associated *status exit* is scheduled as a local SRB in the address space which was current primary when the *join* service was invoked. The *status exit* may perform any actions necessary to determine the correct execution of the member. If the *status exit* returns to MVS indicating that the member is operational, member monitoring continues with no further action. If the *status exit* returns to MVS indicating that the member is not operational, a member-status-update-missing notification event is formulated. If the *status exit* does not return to MVS within three invocations of the system monitor from the external timer interrupt, MVS processing declares the member to be not operational, and a member-status-update-detected-missing notification event is formulated. Member-status-update-detected-missing events are delivered to the *group exit* of all members in the group.

If a member-status-update-missing condition is detected, monitoring of the member does not stop. Member monitoring continues to schedule the *status exit* to determine member status. Monitoring continues until the member resumes execution or terminates. A member is determined to have resumed execution when the *status exit* reports correct execution of the member. When a member resumes correct operation, a member-status-resumed condition is reported to the *group exit* of all other members in the same group.

There exists a relationship between the system monitor and member monitoring regarding when a member-status-update-missing event is initiated by MVS coupled-systems services. If no member monitoring is in effect, a system-status-update-missing event is reported as soon as the condition is detected. If member monitoring is in effect, a system-status-update-missing event is not reported until the member-specified time interval has expired.

Events which cause the loss of point-to-point signaling connectivity between any two systems in a sysplex result in the removal of a system from the sysplex. Loss of connectivity between systems occurs only when the last path used to send signals between a pair of systems fails. When the last path fails, an operator prompt is generated to allow for full signaling connectivity to be reestablished. A loss-of-signaling-connectivity event occurs if full connectivity cannot be reestablished. When loss of signaling connectivity occurs, the operator is required to specify which of the two systems that can no longer exchange signals is to be removed from the sysplex. The system to be removed must be reset, and the operator must indicate when the system reset has completed. Notification of a system having been removed from the sysplex is provided to the *group exit* of all members.

All systems in the sysplex must be time-synchronized. In a configuration comprising multiple central processing complexes (CPCs), synchronized time is provided by the IBM 9037 Sysplex Timer™, otherwise referred to as the *external time reference* (ETR). The ETR can be configured such that there exist two interconnected IBM 9037 Sysplex Timers and duplicate connections to the Sysplex Timer from the CPCs for expanded availability. Should a system lose the services of the ETR, either that system must be removed from the sysplex or all other systems must be removed from the sysplex. Loss of ETR services could occur because of failures in the attachment of a CPC to the ETR or because of an ETR failure. If ETR services are lost and the affected system is part of a multisystem sysplex, MVS services place the system in a restartable wait state. If ETR services can be reestablished for a system which has been placed in a restartable wait state, a system restart restores the sysplex configuration. If ETR services cannot be reestablished, and a system is restarted from the wait state, all other systems must have been reset and therefore removed from the sysplex. Notification of a system having been removed from the sysplex is provided to the *group exit* of all members.

In addition to the events identified above, processing to remove a system from a sysplex may be initiated by operator command. As in other cases, the system being removed from the configuration must be reset by the operator. When the operator indicates that the system reset has been performed, members which remain in the sysplex are notified of the event through the *group exit*.

• *Signaling services*

The signaling service provided by MVS coupled-systems services has as its primary objectives

- To support a datagram form of communications between members of a group.
- To provide the highest performance possible by minimizing all delays in message delivery and minimizing processing overhead by having the shortest possible instruction path length.
- To make available a base system service with a durable interface which can take advantage of future architecture and hardware facilities.

The signaling service enables cooperating instances of a function to communicate within a sysplex. Communications outside the sysplex are managed by network products. Because the signaling service supports members within a sysplex, the robust function provided by a general network server is not required. The inherent cooperation between two instances of a function within a sysplex and the use of highly reliable connection hardware between systems within a sysplex remove requirements for enforcement of communications protocols within the signaling service. Because the signaling service need only provide a simple datagram function, the instruction path length can be minimized. Performance is further enhanced by making highly optimal use of the underlying hardware facilities. Maximizing the signaling performance enables more frequent signals to coordinate functions among members within a sysplex and to thereby improve the single-system image characteristics of a sysplex.

The signaling service provides a highly reliable mechanism for delivery of messages between members. In all cases except storage overlays, signals are delivered or notification of member or system termination provided. However, if programs require knowledge that a signal has been delivered and processed by the receiving program, it is the responsibility of that program to provide acknowledgment protocols.

The external interface to the signaling service is the same regardless of where within the sysplex the sending and receiving members are executing. The *message out* service requires specification of the sending member, the receiving member, and the location and size of the signal. Optionally, a small control area may be specified. The parameters and signal are copied into formatted buffers managed by MVS coupled-systems services before control is returned to the caller.

Signals are delivered to the receiving member through the *message exit* specified to the *join* service. The *message exit* is scheduled as a global SRB to the address space which was current primary at the time the *join* service was invoked. The *message exit* is scheduled once for delivery of each signal and may be scheduled in parallel to deliver multiple signals. MVS coupled-systems support code is given control before and after execution of the *message exit*. A parameter list is provided to the *message exit* which identifies the sending member, contains the control area, provides a token which represents the signal, and provides the size of the signal. If the receiving member is to process the message, the token value from the parameter list must be provided to the *message in* service. Additionally, a location in which to place the message must be provided. If the receiving member does not process the message, MVS coupled-systems support discards the signal when the exit returns control.

Delivery of a signal to a member which is executing on the same system is accomplished by scheduling the *message exit* directly. The maximum amount of buffer space to be used to contain signals awaiting delivery to members on the same system may be specified at system initialization and may be changed during ongoing operation of the sysplex through operator commands.

Delivery of a signal to a member which is executing on a system other than the system on which the sending member is executing requires that the signal be sent over a channel-to-channel (CTC) connection. The term CTC includes, but is not restricted to, devices such as the IBM 3088 MCCU and the IBM ESCON™ Channel.

Each CTC path which is to be used by MVS coupling services must be identified either at system initialization or during ongoing operation of the sysplex through operator commands. Definition of a CTC for use by signaling requires specification of the device number. Definition of a CTC may optionally specify the maximum amount of buffer space to be used to contain signals awaiting delivery across the CTC and the number of errors which may occur before the path is determined to be not operational. The buffer areas used for messages between systems are fixed in real storage and contain formatted channel program segments in order to minimize the instruction path length for sending a signal.

During initialization of a CTC, specialized protocols are used to ensure appropriate interconnection between systems and to initialize values which are used during mainline operations to determine the successful operation of the CTC. Appropriate interconnection between systems is determined to be a connection between systems in the same sysplex and use of the CTC solely by MVS coupling services. Checking for systems in the same sysplex is accomplished in part by comparing the sysplex name, system name and status, and coupling data set usage. A key element in determining the correct execution of the CTC for mainline operations is knowing the maximum number of inbound signal areas that may be chained together in a channel program at any one time.

Use of the underlying CTC hardware for signal delivery is tailored to achieve the best possible performance. One technique used to improve performance is to allow signals to flow across any single CTC in one and only one direction. The CTC paths used by MVS coupling services must therefore be defined to be either inbound paths for a system or outbound paths for a system. The minimum configuration requires one inbound and one outbound path between each pair of systems in the sysplex. Any number of inbound and outbound paths may be specified. MVS coupled-systems services use all paths made available in parallel in order to achieve the lowest latency in delivering a signal and to achieve the highest bandwidth possible, given the available hardware configuration. By directing signals in a single direction, the amount of hardware and software protocol overhead for managing the direction of data transfer is minimized.

The channel program segments for delivery of a single message are also optimum for performance. Each channel program segment comprises a prepare, either a read or write, and a no-op channel command word. The read and write channel commands use indirect address words to locate the signal data. The System/390 I/O architecture constructs which support seldom-ending channel programs are used in conjunction with suspend and resume protocols to eliminate overhead when initiating an I/O operation to the CTC. The no-op channel command word carries the suspend flag. When a new signal is to be sent over the CTC, the no-op is changed to a transfer-in channel. If the channel program is suspended, a resume-subchannel command is issued. If the channel program is not suspended, it is assumed that the no-op has not been fetched by the channel, and therefore the transfer-in channel will operate correctly. If this assumption proves to be incorrect, an I/O interrupt occurs and the I/O is restarted via the resume-subchannel instruction.

A suspend interrupt is enabled which can be used to measure the progress of execution of the channel commands on the CTC. However, in conditions of heavy signaling traffic, the suspend interrupt rarely occurs, since the no-op is often changed to chain to the next signal. Therefore, program-controlled interrupt (PCI) processing is exploited to detect progress through long chains of channel programs. As part of the disabled processing associated with the PCI I/O interrupt, signals are placed on a queue used to support retry should the CTC fail.

Errors may be reported by the channel subsystem through System/390 architected conditions. Error recovery may determine that a failure is permanent or temporary. For permanent failures, the path is no longer used by MVS coupled-systems signaling support. Software protocols are used over correctly operating paths to determine which signals were successfully delivered and which signals scheduled for delivery over the failed path must be retried on paths which continue to operate correctly. For temporary errors, the failing path is restarted. A count of temporary errors is incremented when a temporary error occurs and decremented when successful operation of the paths is determined. A retry value may be specified by the installation or a default value used to determine a threshold for temporary errors which cause the path to be considered permanently failed.

MVS coupling services for signaling rely on I/O subsystem characteristics and software protocols to implement error detection and correction for CTC usage. Errors may be detected by the CTC hardware and reported through System/390 architected conditions. Errors may also be detected by software protocols, which include comparing headers and trailers with each signal to detect completeness and use of sequence numbers with each signal to ensure that all signals are delivered over a CTC path. Underlying recovery for software-detected errors is the System/390 I/O architected ability for a system to initiate a halt/clear signal to cause the matching channel program on the other side of the CTC device to end in error.

Software protocols implemented by the sending and receiving systems are employed to enhance the basic error-reporting mechanisms provided by the hardware. The sending system knows the maximum number of inbound signal areas that may be chained together in a channel program at any one time. As PCI or suspend interrupts are processed to reflect the progress of the sending system's channel program, signals are placed on a queue which supports retry operations. As PCI or suspend interrupts are processed to reflect the progress of the receiving system's channel program, the completeness of the signal and correctness of the sequence number are verified. If an error is detected on the receiving side, no new channel program segment is added to the receiving side. Therefore, at any one instant in time, the maximum number of signals which may have been sent but not yet successfully received is equal to the maximum number of inbound signal areas. On the sending side, the number of signals on the retry queue need only match the number of inbound signal areas. When this protocol is used, no acknowledgment signals are required. If the receiving side detects an error, it initiates recovery through the halt/clear sequence.

When a sending or receiving channel program ends in error, a software protocol is initiated to restart the path. Special signals and protocols are employed to determine what signal sequence number was last successfully processed by the receiving system. All signals scheduled for delivery by the sending side which were not successfully delivered are re-sent over operational paths.

A second class of signal delivery failures occurs if signals are being sent but not received. The signal delivery timer support for signaling paths is given control from MVS coupled-systems monitoring support. For each signaling path, the signal delivery timer determines whether signals scheduled for delivery over a path are being delivered to the target system. If signals are not being delivered, the failing path is restarted. As part of the restart, all unsuccessfully delivered signals are re-sent on operational paths.

MVS coupled-systems support provides the ability to define transport classes for signaling traffic. Transport classes are used to segregate signals being sent into discrete, named classes. Through the use of transport classes, an installation can tune the performance of signaling traffic within a sysplex by controlling the competition among programs for signaling resources. Transport classes provide the flexibility to segregate message traffic according to application, message length, or both. A transport class is defined by a name, a signal length for which the class is optimum, the maximum amount of buffer space to be used to contain signals awaiting delivery in the class, and the set of group names associated with the transport class. Each outbound signaling path may then be associated with a transport class. During normal operation, signals sent from members of groups associated with a transport class are delivered over the paths associated with that transport class.

A group is usually allowed to use only the signaling resources associated with the transport classes to which it has been assigned. A group is allowed to use signaling resources in transport classes to which it is not assigned only when there are no operational paths in any of the transport classes assigned to that group. The signaling service automatically routes signal requests over an alternate transport class when there exist no paths to service the transport class to which a group is assigned.

• *Time services*

All systems within the sysplex are required to be time-synchronized. Synchronized time provides a basis for correlating events between systems to achieve a closer single-system image. Events which may require correlation include system logs; dumps, traces, and logs for serviceability; performance monitoring; and capacity planning data. Additionally, synchronized time permits events to be correlated in real time. Resolution between events on different systems must be more granular than the length of time it takes a signal to propagate between systems and more granular than the length of time it takes for access to any shared data.

Sequencing between events on different systems could have been accomplished with a global sequence number generator. However, a global sequence number generator introduces a response time penalty and can become a capacity constraint. With a global sequence number, two signals, one to the facility and one in return, would be necessary. The speed of a signal is limited by the fastest available link technology. Generation of a time stamp from a synchronized clock performs much better and meets the granularity requirement.

During system initialization, use of the ETR is controlled by installation specification. An installation may choose to have processor time-of-day clocks set automatically from the ETR without operator intervention. If an ETR is being used, the installation may specify whether the system is to get the time zone constant from the ETR or use a system-specific value. The installation may also specify the greatest difference between the system time of day and the ETR which will be automatically corrected.

For configurations comprising systems in a sysplex executing under VM/ESA® or within one CPC under the IBM Processor Resource/Systems Manager™ (PR/SM™), a simulated ETR may be used to achieve clock synchronization. The identifier for a simulated ETR may be specified by the installation to the system upon initialization. Systems which use a simulated ETR cannot become part of a sysplex that uses multiple CPCs.

Programming interfaces are provided to obtain a clock value. Through the architected System/390 STCK (Store Clock) instruction, a program can obtain a clock value. If the system clock is synchronized with the ETR, the value obtained is a sysplex-synchronized clock value. For programs which require knowledge of whether or not the value obtained is synchronized with the ETR, the *STCKSYNC* macro provides a high-performance interface to return a clock value and an indication of ETR synchronization.

• *PR/SM capabilities*
PR/SM, which divides a processor into logical partitions, now includes an automatic reconfiguration facility (ARF). The ARF function provides a platform for those applications that have availability requirements. ARF consists of two systems: the primary system, which runs the application, and the backup system, which provides application services in the event of a primary system failure. ARF, when used in conjunction with an automation product, allows workload redistribution from the failed primary system to the backup system without operator intervention. ARF can be used in either a single-CPC environment or a multiple-CPC environment, depending on availability requirements.

The single-CPC environment, where the primary system and the backup system are executing in logical partitions on the same CPC, provides high availability against MVS system failures. The primary logical partition, defined to have the majority of the CPC resources, runs the production workload while the backup logical partition, defined to have minimal CPC resources, monitors the primary system. When the backup system detects that the primary system appears to be not functional, processing of the MVS coupled-systems services PR/SM policy is initiated. This policy provides the ability to specify that the backup system is to reset the primary logical partition, deactivate the primary logical partition, and acquire the storage resources defined to the primary logical partition. ARF uses the PR/SM dynamic storage reconfiguration facility for the backup logical partition to acquire the storage resources previously owned by the primary logical partition.

The multiple-CPC environment, where the primary system and the backup system are executing on different CPCs, provides high availability against software and hardware outages. Specifically, this can protect against MVS system failures, hardware failures, and application failures. The primary system, executing in basic mode or in a large logical partition, runs the production workload on one CPC, and the backup system, running in a small logical partition on another CPC, monitors the primary system. The backup system, when it detects that the primary system appears to be not functional, implements the MVS coupled-systems services PR/SM policy. This policy provides the ability to specify that the backup system is to deactivate expendable logical partitions and acquire the storage resources defined to them. ARF uses the PR/SM dynamic storage reconfiguration facility for the backup logical partition to acquire the storage resources previously owned by the expendable logical partition.

• *System services characteristics*
MVS coupled-systems services utilizes a shared data-set structure to maintain status information regarding systems, groups, and members. The coupling data set must be directly accessible and is shared by all systems in the sysplex. It supports monitoring services by providing the basic mechanisms for recording system status outside of the system and retrieving the status of other systems within the sysplex. The coupling data set supports group and member services by providing nonvolatile recording media for permanent status recording. Services which access the coupling data set provide for high availability and serialized access to data while avoiding single points of failure or delay between systems.

The coupling data set must be created and made available for use through installation definition. A utility is made available to determine the size of the coupling data set and format the DASD records for subsequent use. The format utility accepts as input a specification of the volume on which the data set is to reside and the capacity required in terms of maximum groups and members in groups. A data set which is successfully allocated and formatted by the utility may be placed in use through specification at system initialization or by operator command.

Correct operation of the services which access the coupling data set is required for systems to exist in a multisystem sysplex. Therefore, a wide variety of services and programming techniques are employed to ensure the highest availability for coupling data-set functions. Services which access the coupling data set support the ability to have a primary and an alternate data set. Data are first successfully recorded on the primary, then recorded on the alternate. Should the primary coupling data set become unusable, services which access the coupling data set automatically switch to the alternate without loss of data. An alternate coupling data set can be made available for use during ongoing operations of the sysplex. Data from the primary coupling data set are propagated to a newly provided alternate coupling data set without loss of access to the primary. Recovery is provided for individual records on the coupling data set. Services which access the coupling data set restore data on the primary from the alternate if required. Furthermore, if required, channel programs which format tracks of the coupling data set are used to restore correct content of the coupling data set. These recovery operations occur transparently with respect to other MVS coupled-services components which use the coupling data sets.

Access to the records on the coupling data set is serialized by a software protocol. Records which can be serialized have associated with them records which are used for software locking purposes. When serialized access to a record is required, services which access the coupling data set record interest in the record and retrieve the current contents of other records which reflect the interest of other systems. A protocol is invoked with this body of data which determines ownership of the lock protecting the serialized record. When a serialized record is updated, atomic checks are performed through channel programs, which ensure that the lock for the record is held by the system performing the update. If a system holds a lock on a serialized record for an excessive amount of time, and other systems are waiting for access to that record, a lock-stealing algorithm is invoked. The lock-stealing algorithm ensures that updates to the serialized record on both the alternate and the primary coupling data sets are blocked for the system which holds the lock. Lock stealing then makes the record available to the oldest waiting process. When an attempt is made to update a record for which a lock has been stolen, the update fails. The process from which the lock was stolen may reinitiate the update by gaining serialized access to the record. Processing for updating serialized records on the coupling data set is inherent in the design of MVS coupled-systems services.

Services which access the coupling data sets provide access capabilities to other MVS coupled-systems services functions. These capabilities include the ability to read a record without serialization, gain serialization and read a record, check serialization and update a record, and release serialization of a record without update.

MVS coupled-systems support provides serviceability aids to assist in diagnostic activities related to a sysplex. Component trace data are maintained as a default, and support is provided for detailed tracing of specific functions. Detailed tracing may be enabled at system initialization time or by operator command. Component trace records may be externally recorded through the component trace external writer. Diagnostic data captured through unformatted system dumps can be viewed through Interactive Problem Control System (IPCS) dialogs and panels. Summary and exception reports are available through IPCS, as well as the ability to obtain a logical view of the control structure and the flow of MVS coupled-systems support.

Performance tuning and capacity planning support are also provided by MVS coupled-systems services. Statistics are maintained during ongoing operation which reflect utilization of signaling services by members and utilization of CTC connections on a path basis. Performance-related information is made available to performance monitors through the *measurement-gathering* service. Reports are made available through RMF which reflect usage of MVS coupled-systems services.

Coupled-systems functions

- *Global resource serialization*

Global resource serialization (GRS) is a component of the MVS operating system which provides general-purpose, logical-lock manager functions. GRS provides serialziation services for application programs, system components, and subsystems. Resources serialized by GRS have either single-system or multisystem scope. The scope of a particular resource serialization action is set initially as a parameter by the program which invokes GRS services. GRS supports a set of resource name lists (RNLs) which can change the scope of a resource serialization request without modifying the requesting program.

Serialization state information is kept by GRS in a data structure replicated on all the cooperating systems. Updates are passed among systems arranged in a ring structure through messages. Each system holds the update message for a period of time, during which updates from other systems are observed and new requests from the system holding the message are added. Prior to MVS coupled-systems support, GRS directly managed CTC connections to support this message passing. The frequency with which updates pass between systems was determined by a fixed time value established either through a default or by installation specification.

In MVS/ESA SP Version 4, changes are made to GRS to utilize MVS coupled-systems services. GRS uses the

MVS signaling services instead of directly managed CTC connections for passing messages. Since MVS coupled-systems services support multiple paths between systems, loss of a single CTC no longer results in a disruption of GRS message passing and associated recovery scenarios.

RNLs are a critical control specification which influences the performance of the system as a whole and determines the effectiveness of serialization requests for data integrity. RNLs must be changed in several circumstances, including adding new applications and execution of work with specific serialization requirements. When RNLs change, all systems in the configuration must synchronize the change in order to ensure data integrity. Prior to MVS/ESA SP Version 4, changes to RNLs could be achieved only through initialization of all systems in the complex. With the new support, RNLs can be changed by operator command. By using the signaling and state-manipulation facilities supported by MVS coupled-systems services, GRS maintains integrity across these changes by suspending any unit of work which requests any resource affected by the change in RNLs until the change is completed. An RNL change may be completed by being canceled by the operator or by being processed by all systems. An RNL change which affects a resource for which serialization requests have already been processed is delayed until all serialization requests for the affected resource are released.

The time span during which the serialization message is kept in one system until it is sent to the next system in the complex is called the RESMIL (residency milliseconds) time. With MVS/ESP SP Version 4, the installation continues to specify a RESMIL time. However, the system can dynamically extend the specified time in increments up to a system-defined maximum. MVS increases the time during periods of low activity in order to increase the probability that additional serialization requests will be added to the message at each system. The residency time is reduced to as little as the installation-specified RESMIL value during periods of high activity in order to improve responsiveness.

Operator actions to recover from disruption in the delivery of the GRS serialization message are complex at the best of times and are very error-prone under the pressures which typically accompany a failure. By using the monitoring and signaling facilities provided with MVS coupled-systems support, systems in a sysplex will automatically rebuild a GRS message ring. Additionally, a system which has been temporarily removed from the GRS message ring will automatically rejoin the ring once it has recovered.

• *Operation services*

The operation services component (OPS—comprising Master Scheduler and CommTask) of MVS provides system support for programs to communicate with the operations staff and for the operations staff to control and monitor the sysplex with system commands. MVS Operator Consoles, typically input/output-capable display-type devices of the 3270 family, are used to carry out these communications. Through MVS coupling services, OPS establishes awareness and communication among all instances of OPS within the sysplex. OPS uses these services for the delivery of messages, commands, and control information among the systems in the sysplex.

The configuration of the MVS Operator Consoles can range from one to ninety-nine consoles for the sysplex. The consoles can be physically attached to any system in the sysplex. One console is required, and that console is called the "master console." The master console is supported by MVS as the primary point of operational control and offers a greater level of function than the other MVS Operator Consoles. MVS also provides a programmable console interface comprising three macros: *MCSOPER*, *MCSOPMSG*, and *MGCRE*. With this interface, applications can be created that provide presentation services similar in function to the MVS Operator Consoles. Exploiters of the MVS programmable console interface are referred to generically as extended consoles.

The MVS Operator Console configuration, including extended consoles, spans the entire sysplex. This means that each system in the sysplex has full knowledge of the console configuration. This knowledge is achieved by replicating the control structures that define that configuration. The replication is done with a distributed shared-data technique which employs signaling for control structure distribution and user state updates to synchronize changes to the shared data. The user state field contains an indicator of the current level of the sysplex shared data. This shared-data technique is used every time a state change to the console configuration occurs.

The following sequence of events identifies the principal activities of the distributed shared-data update:

1. Begin serialized shared-data access:
   a. Acquire "lock" (a global GRS resource serialization request).
   b. Check the local shared-data state with the sysplex shared-data state. This is done by comparing the local shared-data level, contained in a local copy of the user state field, with the sysplex shared-data level, which is contained in the user state field in the coupling data set.
   c. If the local shared-data level is inconsistent with the sysplex shared-data level, then acquire a copy of the shared data from a system possessing the most current level.
2. Update local copy of shared data.

3. Update other systems in sysplex:
   a. Package updated shared data.
   b. Distribute package(s) to other systems in sysplex via signaling.
   c. Wait for acknowledgments of receipt from all other systems (acknowledgments are also sent via signaling) or timer expiration, whichever occurs first.
   d. Increment sysplex shared-data level by updating the user state field in the coupling data set to signify that the update has been committed. This update causes the other systems to be notified via group services. Systems receiving this notification apply the shared-data update to their own local shared-data copy. Systems unable to do so acquire the current level of the shared data from another system during the next shared-data update or as a result of periodic shared-data-level monitoring, performed on each system.
4. End serialized shared-data access.

One ramification of the replicated control structure is that any console can select any message generated by any system in the sysplex. The messages are distributed via signaling. Controls are available for both MVS Operator Consoles and extended consoles to specify message selection criteria. Examples of these controls include routing codes (ROUT) and message levels (LEVEL). While many of these controls are useful even in the single-system environment, one particular control is unique to the coupled environment because it permits the system on which a message originated to be specified as a selection criterion. This control is called Message Scope (MSCOPE) and it allows a console to select messages that originate from one, several, or all systems in the sysplex. All other message-selection controls specify criteria that filter messages from the set selected by MSCOPE.

A further ramification of the replicated control structure is that any console can control and monitor any resource in the sysplex through the use of system commands. With the initial release of coupling, the command set is divided into two subsets:

1. *Commands with sysplex scope*
   Commands with sysplex scope enable resources within the sysplex to be managed without explicit knowledge of the individual systems on which those resources reside. This subset of commands includes the following:
   - *Commands that control consoles*
     These commands are used to control the characteristics of the MVS Operator Consoles and extended consoles. An example of these characteristics is the message-selection criteria mentioned previously. This means that a command, such as VARY CN(MASTER), MSCOPE = SYS2A, which alters the MSCOPE of the console named MASTER, can be successfully processed even when the command is entered on a system different from the one on which console MASTER is attached.
   - *Commands that manage program and system requests*
     These commands permit the list of outstanding requests to be viewed by any console in the sysplex. The list contains all outstanding requests in the sysplex. These requests are in the form of messages, and are maintained in the Action Message Retention Facility (AMRF). Operator prompts (called WTORs), a type of request, can be responded to from any console in the sysplex (subject to authorization). This means that a WTOR issued on a given system in the sysplex can be viewed via the DISPLAY R command and answered from any console. The issuer of the REPLY command need not be aware of which system the WTOR originated from because the reply ID on the WTOR is unique within the sysplex. The unique reply IDs enable the REPLY command to be internally routed to the system from which the WTOR originated to complete its processing.
2. *Commands with system scope*
   These commands require explicit knowledge of the systems in the sysplex. The user of these commands must determine the system to which the command must be directed. OPS provides services to facilitate the delivery of commands to specific systems. Signaling is used to transport the commands to their intended destinations. The following is provided:
   - *ROUTE command*
     This command permits a single command to be directed to a specific system.
   - *Persistent command routing*
     A console control is provided that permits all commands issued from a console to be automatically routed to a specific system in the sysplex. The control permits the persistent routing to be assigned to another system dynamically. The console control for this capability is called command association (CMDSYS). Through CMDSYS the system to which the commands are to be delivered for processing is specified.
   - *Command prefixing*
     This is a programmable interface that enables a character string to be associated with a particular system in the sysplex. The interface is provided by the *CPF* macro. When a command is issued with this character string as a prefix, the command is automatically directed to the associated system. The registration of these prefixes is known to all systems in the sysplex through the replicated control structure. This facility can be used to provide sysplex scope for subsystem commands. MVS subsystem commands (e.g., JES, DB2®) normally begin with a command

prefix unique to the particular subsystem. Subsystem commands directed to subsystems that have registered their prefix with the sysplex can be issued from any console in the sysplex and are automatically transported to the system on which the target subsystem resides.

• *OPC/ESA*
The current environment of processor consolidation and continued growth has created increasingly significant demands on availability. This includes such items as the production workload and a shrinking batch window on off-shift hours. Elimination of manual scheduling and automation require a high level of event management. Elements necessary to this process, while present in several products, must now be able to operate in an MVS sysplex.

IBM Operations Planning and Control/ESA (OPC/ESA™) provides centralized control for submission of jobs, resolution of job dependencies and interrelationships, monitoring of job flow in the sysplex, and automatic workload recovery for the MVS/ESA system. In a sysplex, OPC/ESA distributes the production workload and tracks job completion status via MVS coupling services. Job return codes, step codes, and abends are examples of the type of feedback information processed. Job Control Language (JCL) is also sent using signaling services, which obviates the need for shared DASD or the use of VTAM® links which were previously used for that purpose.

OPC/ESA is composed of a controller feature and a tracker base. The controller is the focal point for any number of trackers. The tracker is present in each MVS image in a sysplex and is responsible for relaying SMF and JES event data back to the controller. Controller and tracker(s) together form a group using MVS coupling services. If an MVS/ESA system or an OPC/ESA controller fails, a dormant controller will take over the necessary function. In a typical scenario, this "hot standby" capability allows the following:

- MVS coupling services provides notification of failure of a member of the sysplex.
- OPC/ESA logically isolates that processor and submits no new work to it.
- The controller retrieves the status of all work on the failed processor and marks unfinished work as ended in error.
- Automatic workload recovery allows restart of ended-in-error jobstreams on another system in the sysplex.

Through use of global variable subsitution and if-then-else-type procedures, fairly elaborate restart implementations are possible. Simple defaults of restarting a job in place from the first step up to user-implemented application-specific recoveries are all possible.

• *CICS/ESA*
Enhancements can be made to the CICS/ESA™ functions for XRF to exploit MVS coupled-systems services. Existing CICS monitoring functions implement a private protocol for detecting the correct execution of the primary. When the primary is determined to be not operating correctly, CICS utilizes JES services to inquire about the status of the primary. If the JES inquiry does not respond or responds that the CICS primary continues to execute, an operator prompt is issued to determine the status of the primary. CICS uses the *query* service after the existing private protocol determines that the primary may not be operating correctly, and before the JES inquiry. If the response from *query* indicates that the system on which the primary CICS had been executing is no longer in the sysplex, the operator prompt is not issued, and takeover by the alternate continues to completion without operator intervention.

• *TSO/E*
In multisystem environments without MVS coupled-systems services, it is necessary to read the SYS1.BRODCAST data set when a user logs on to TSO to retrieve the most current copy of system notices. TSO/E 2.2.0 utilizes MVS coupled-systems services to maintain the broadcast information in storage on all systems in the sysplex. Whenever changes are made to the notices section of the SYS1.BRODCAST data set, notification of change is propagated to all systems, using signaling services. When updates are received, the in-storage image of the broadcast information is updated, thereby eliminating the need to read the data set for every logon and eliminating I/O device contention caused by accesses to SYS1.BRODCAST during heavy user logon periods.

• *SLIP*
When a problem is detected by a Serviceability Level Indication Processing (SLIP) trap on one system, some of the serviceability data needed to solve the problem may reside in one or more other system(s). Therefore, SLIP provides a way for a SLIP trap on one system to initiate an SVC dump or load a wait state on another system. This function can be used in a sysplex for second-failure data capture when analysis of a previous failure has found the need for additional data from one or more system(s) other than the one which detected the problem.

A new REMOTE keyword is provided on the SLIP command to specify actions to be taken on systems other than the system on which the trap matches. The REMOTE keyword specifies the system (as a system name or Group.Member), the action (WAIT or SVCD), and the dump options if the action is SVCD. The REMOTE keyword contains descriptions for all of the actions to be taken on all of the other systems. The action on the system where the trap matches continues to be specified by the existing ACTION keyword.

The REMOTE keyword causes the SLIP action processor to schedule an SRB which uses the *query* service to translate group/member pairs into identification of systems and address spaces. The SRB then sends a signal to SLIP on each of the requested systems passing the action (SCVD or WAIT) and the dump options if the action is SVCD. SLIP processing which receives the signal builds the appropriate SDUMP parameter list and invokes dumping services or causes a message to be sent to the system console and a restartable wait state to be loaded.

Future directions

Currently, it is intended that successive stages of MVS coupled-systems support will occur over time and across many products. Where appropriate, new System/390 architecture and hardware support will be provided along with associated base MVS software services. Meeting the requirements arising from the markets addressed by MVS coupled systems will necessitate support in the following areas:

- Enhancements to data access.
- Workload management improvements.
- Greater numbers of systems in a sysplex.
- Removal of systems management constraints.
- Availability enhancements.

Access to data in an MVS environment is provided by the MVS/Data Facilities Product (MVS/DFP™) and data base managers including DB2 and IMS/ESA®. In addressing the coupled-systems environment, changes will be required in these products to support the single-system image goal for a sysplex and to take advantage of the potential for availability improvements in a sysplex configuration. Providing access to data across the systems in a sysplex while maintaining high availability and a single-system image will require these products to address fundamental multisystem serialization and data caching constructs. Serialization of data must allow access by many concurrent users working on multiple systems within a sysplex at performance levels which are competitive with those of a single system. Caching of data in memory is utilized in existing systems to improve the performance of data access. In the sysplex environment, data caching must continue to be supported in order to obtain equivalent performance. While maintaining data access performance, the data caching structure must ensure the consistency and integrity of data accessed by applications executing across the systems in a sysplex.

Existing workload management in an MVS environment is provided through a variety of products, including the IBM-supplied Job Entry Subsystem (JES), the CICS transaction manager, the IMS transaction manager, and the OPC/ESA production workload scheduler. One of the fundamental challenges facing workload management products is the ability to classify work into categories and associate performance goals with categories of work. In order to effectively utilize the capacity available across systems in a sysplex environment, changes are required which will enable installation specification of work classification rules, definition of performance goals for categories of work in terms of response time and throughput, and specification of overall data processing capacity to be provided to categories of work. These changes move away from existing external parameters which specify rules for distribution of physical resources such as CPU and storage without regard for the overall effect on service-level agreements for delivery of work.

When changes are made to the externals by which the installation defines goals for distribution of data processing capacity, corresponding changes will be required in the underlying MVS components which manage physical resources such as access to CPUs and storage usage. Algorithms will be required which utilize the performance goals defined by the installation, observe the execution characteristics of the work, and adjust the distribution of physical resources to meet installation specifications. Effective distribution of physical resources can best be accomplished by establishing a common understanding of the performance goals between the workload management products and facilities which distribute physical resources. On the basis of a common understanding, monitoring facilities can be used to measure the achieved performance of work and make adjustments to better meet the performance goals of all work.

Work scheduling and backlog management is another key area supported by the workload management products. Within a sysplex environment, the scheduling and backlog management services must be able to receive work from any system, determine the ability of servers of that work across the sysplex to perform new work, and distribute work across the sysplex in a manner which best achieves the performance goals of the work. This function is performed in existing systems by transaction managers such as CICS and IMS and scheduling facilities such as JES and APPC. A variety of techniques are employed, including transaction routing, message queues, and shared work queues. Enhancements for the sysplex environment can be achieved by minimizing the overhead needed to perform multisystem work scheduling and making efficient use of available capacity within the sysplex to meet installation-specified performance requirements.

The ability to provide unconstrained, incremental growth will require support for an increasing number of systems in the sysplex configuration. As the number of systems in a sysplex grows, greater connectivity to devices and among systems is required. The design of software algorithms must provide for efficiencies when the number of systems is small, while still having the capability to support large numbers of systems in the sysplex.

Aspects of installation, customization, operations, performance tuning and capacity planning, and serviceability have sysplex considerations. With increasing numbers of systems in the sysplex, additional systems management capabilities must be provided. In each area, complexity must be minimized and single-system image characteristics for a sysplex enhanced.

The sysplex configuration provides a base on which improvements in availability can be achieved. However, careful attention must be given to common single points of failure in hardware and software design to ensure availability improvements. Steps toward achieving continuous availability can be taken within a sysplex configuration which are not feasible in a single-system environment.

Summary

Basic operating system support for the MVS coupled-systems environment is provided in MVS/ESA SP Version 4. System services are made available and utilized within the MVS coupled-systems environment to improve the availability and systems management characteristics of a configuration comprising multiple MVS images. It is currently intended that these building blocks will form the foundation on which growth and availability requirements can be met through enhancements in future products.

Enhancements may be made to the MVS coupled-systems environment through new System/390 architecture, hardware, base MVS services, and exploitation by a wide range of products. Key areas to be addressed include data access, workload management, a greater number of systems in the configuration, systems management, and availability. Changes made in these areas should improve the overall MVS coupled-systems characteristics such that a single data processing capability is presented.

The facilities provided by the MVS coupled-systems environment are a key element in meeting both immediate and long-term market requirements. The staged introduction which moves the MVS operating system to a multisystem environment has begun and is designed to provide increased value over time. At present, IBM intends the delivery of solutions which scale from small to very large capacities for on-line transaction processing, decision support, data warehousing, and backbone network support to be based on the MVS coupled-systems structure.

Acknowledgments

This paper would not have been possible without the contributions and review of many people. J. Rankin provided an early draft from which material was extracted to create the paper. Material was extracted and edited for the paper from the International Technical Support Centers document number GG24-3628-00, "MVS/ESP SP Version 4 Release 1 Implementation Guide." S. Hart authored the OPC/ESA section. M. Brooks, A. Salverson-Seales, M. Shenhouse, N. Matsakis, and J. Insalaco reviewed and corrected the sections on Base System Services. I. Redding reviewed and corrected the section on CICS. D. Palmieri reviewed and corrected the TSO/E section. D. Ault provided initial input on OPC/ESA, TSO/E, and CICS. T. Monza provided initial input on OPC/ESA. J. Aman reviewed the workload management Future Directions section. R. Allen, D. Ault, F. Manfredo, P. Enichen, S. Puglia, H. Yudenfriend, C. Clark, T. Borden, C. Bernstein, R. Knueppel, and R. Suko reviewed the entire paper for content and correctness. Our thanks go to all of the above for their timely assistance.

Reference

1. *MVS/ESP SP Version 4, Release 1 Implementation Guide*, Order No. GG24-3628-00, January 1991; available through IBM branch offices.

Received May 1, 1991; accepted for publication August 13, 1991

Michael D. Swanson *IBM Enterprise Systems, P.O. Box 950, Poughkeepsie, New York 12602 (D84MDS1 at PLPSC).* Mr. Swanson received a B.A. in physics from the University of Kentucky in 1973. From 1973 to 1979 he worked as a systems programmer for the State of Kentucky. Mr. Swanson joined IBM in 1979 working on early versions of the MVS/XA™ system. Since 1983 he has worked in the MVS design area. His work is currently focused on system structure and services in support of the MVS multisystem environment.

Christopher P. Vignola *IBM Enterprise Systems, P.O. Box 950, Poughkeepsie, New York 12602 (VIGNOLA at POKVMCR3).* Mr. Vignola received a B.S. in computer science from the State University of New York in 1985. From 1984 to 1985 he participated in the IBM Cooperative Education Program, working on distributed archival systems. Mr. Vignola joined IBM as a regular employee in 1985 working on the design and implementation of operational support for the MVS/ESA system. Since 1990 his work has focused on the application of object-oriented technology to large systems and its use in re-engineering legacy code.