PC/SC Technical Overview

Presented at CardTech/SecurTech

CP8 Transac, A Bull Company

Hewlett-Packard Company

Microsoft Corporation

Schlumberger SA

Siemens Nixdorf Information Systems, Inc.

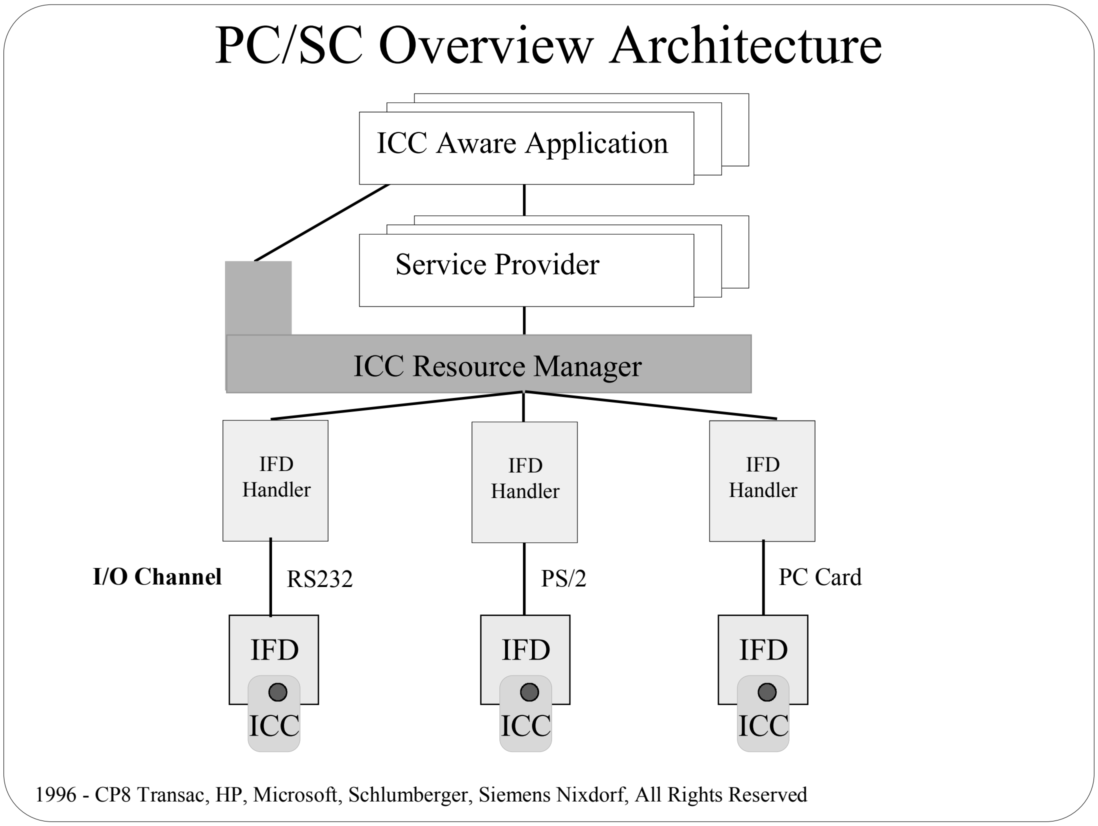
-

December 1996

Presented by: Philippe Sarlin,

Project Manager, HP

PC/SC Overview Architecture



1996 - CP8 Transac, HP, Microsoft, Schlumberger, Siemens Nixdorf, All Rights Reserved

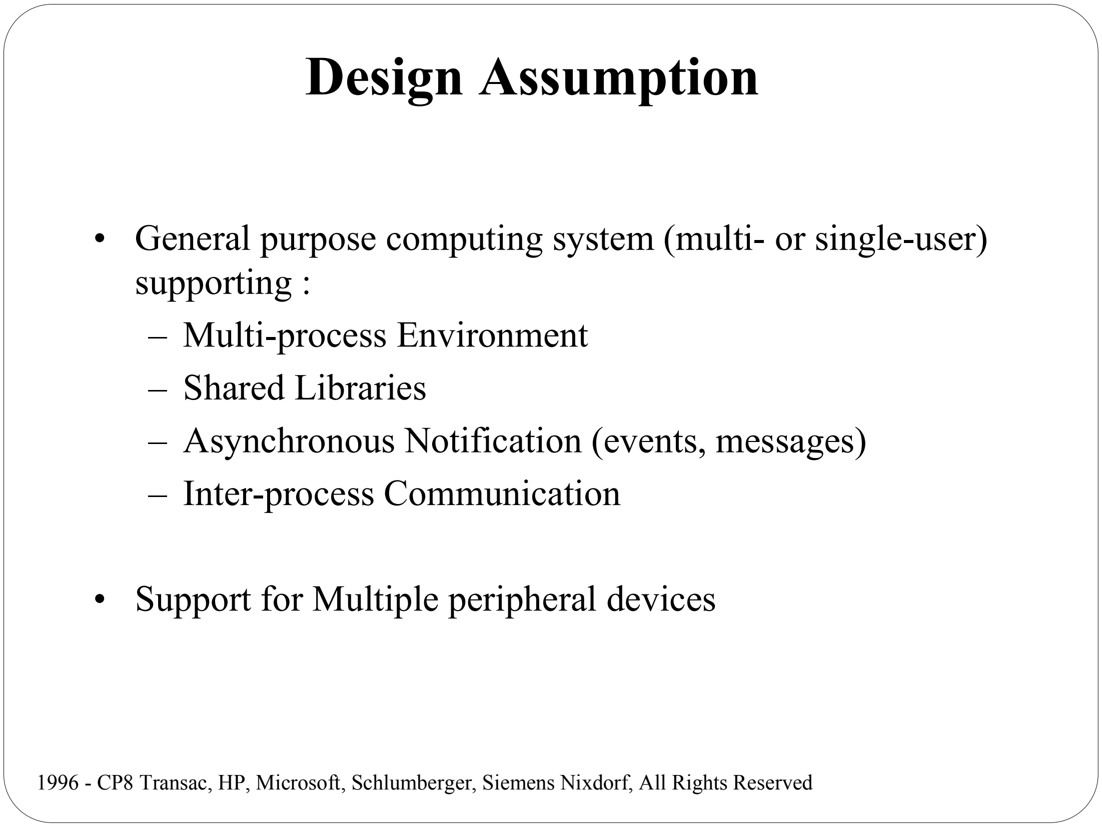
The goal of the PC/SC Architecture is to allow PC based applications to utilize functionality provided by one or more specific Integrated Circuit Cards (ICCs). It has been developed to allow this to be done in a very flexible manner which will support both existing and future ICC-based applications.

The PC/SC Architecture provides a framework within which ICCs (often referred to as Smart Cards) can be utilized on a general purpose PC, It is designed to support a high degree of vendor independence through clean separation of components and interface definitions which promote interoperability amongst products. Specifically, applications can be developed, requiring a specified functionality set, which may be provided by ICCs from a variety of vendors. Moreover, the physical Interface Devices (IFDs), or readers, which provide access to an ICC through standard PC “I/O ports” may be provided by a variety of vendors as well.

The defined architecture clearly segregates those components that would be provided by an Operating System vendor, IFD vendor, and ICC vendor and promotes independent development.

The components defined by the PC/SC Architecture include:

- The IFD Handler that is an installable component provided by IFD vendor.
- The Service Provider that is an installable component provided by ICC vendor This is actually defined as two complimentary components in order to isolate cryptographic functionality due to existing import/export requirements.
- The ICC Resource Manager that is an Operating System component provided by the OS vendor.



Design Assumption

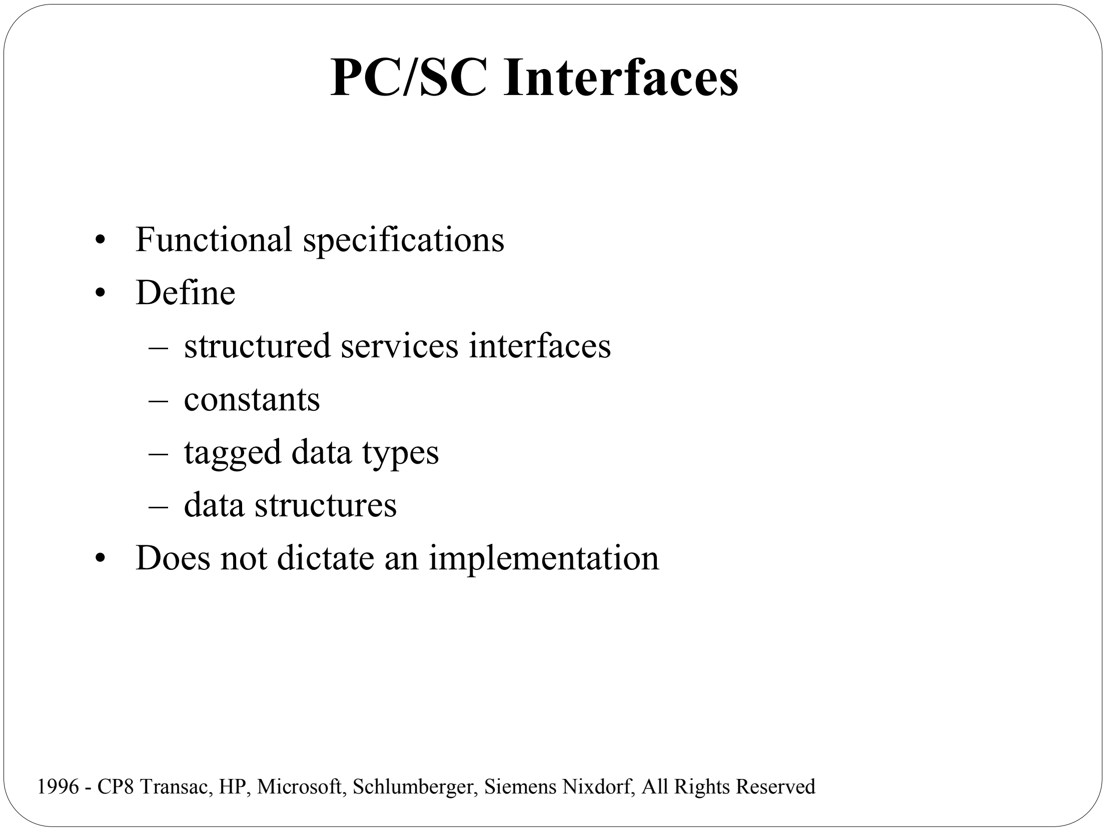
- General purpose computing system (multi- or single-user) supporting :
  - Multi-process Environment
  - Shared Libraries
  - Asynchronous Notification (events, messages)
  - Inter-process Communication

- Support for Multiple peripheral devices

The Workgroup focused its efforts on creating an infrastructure for integrating ICCs with general purpose computing environments. It was assumed that the following services are available:

- Multi- process Environment: Several processes run in a concurrent manner without any effect from one to other.
- Shared Library: It is a shared code module, loaded and linked by a process that uses it.
- Asynchronous Notification: An independent process informs another independent process at any time by sending it a message or generating an external event to the process.
- Inter-process communication : It is a communication and synchronization mechanism among 2 independent and concurrent processes .

These assumptions are appropriate to a variety of computing environments. It includes existing platforms commonly referred to as "PCs" such as Windows and the Macintosh, but is equally applicable to Unix workstations, and emerging platforms such as the NC or Java machines. The Workgroup intends the term PC to be interpreted in the broadest sense.

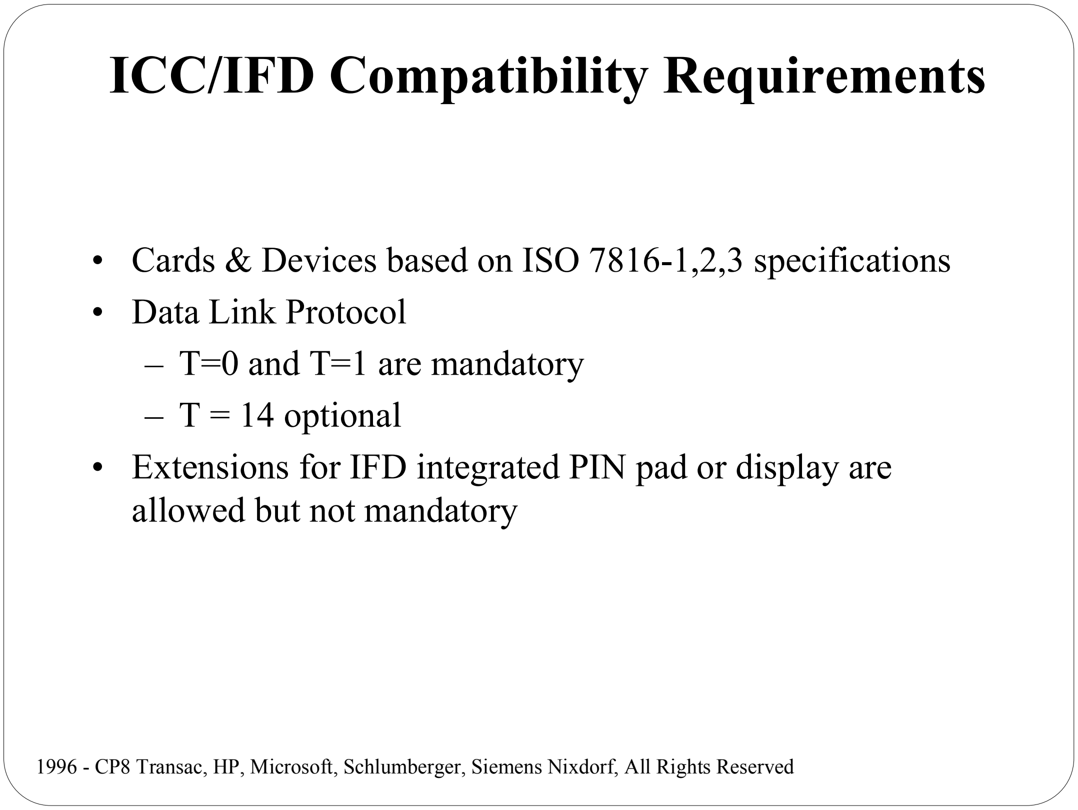


PC/SC Interfaces

- Functional specifications
- Define
  - structured services interfaces
  - constants
  - tagged data types
  - data structures
- Does not dictate an implementation

The Workgroup has focused on the specification of component model with well-defined interfaces. These interfaces are defined in terms of required functional behavior. To promote consistency between implementations we further describe this functionality in terms of methods which expose the required services along with key parameters, supporting constant values (status codes, defined operational modes, …), tagged data types (vendor information, enumeration of capabilities) , and complex data structures where required.

This approach leaves flexibility for system implementers to adapt to the requirements of specific systems. For example, systems vary substantially in terms of API conventions, supported programming languages, IPC and signaling mechanisms, etc. It is our intent to allow implementers to tailor the defined interfaces to meet the requirements of their environment, so long as the required functional behavior is supported.



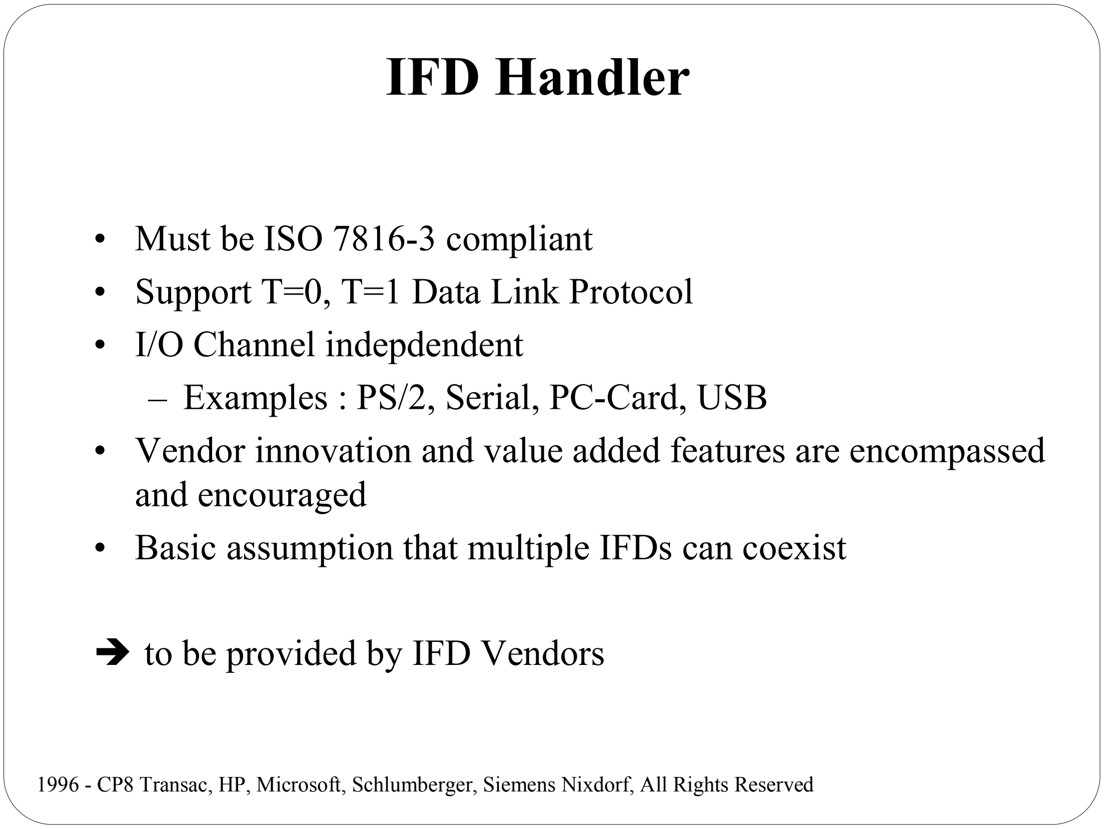
ICC/IFD Compatibility Requirements

- Cards & Devices based on ISO 7816-1,2,3 specifications
- Data Link Protocol
  - T=0 and T=1 are mandatory
  - T = 14 optional
- Extensions for IFD integrated PIN pad or display are allowed but not mandatory

These specifications are intended to support ICCs and IFDs that are compliant with the ISO 7186-1, -2, and -3. The Workgroup specifications indicate a specific compatibility profile. A principle requirement is the ability of the ICC to respond with an Answer To Reset (ATR) sequence using asynchronous answer to reset.

At the data link layer, both the T=0 and T=1 protocol are directly supported. It is also possible to communicate with T=14 devices, but associated Service Providers will need to incorporate support for the specific protocol. The basic interface structure has been developed to make it easy to add support for additional protocols in the future.

Finally, we recognize the desire for some applications to isolate entry of card holder verification data (PINs), or provide an isolated display of information sent to the ICC. Implementations of this functionality require specific knowledge of the application and ICC implementations to be supported. Hence, we did not mandate such functionality, or define standard interfaces for supporting it. Mechanisms are available to make use of such functionality but the Service Provider and/or application will need to be aware of devices meeting their requirements and the means of invoking that functionality.



IFD Handler

- Must be ISO 7816-3 compliant
- Support T=0, T=1 Data Link Protocol
- I/O Channel indepdendent
  - Examples : PS/2, Serial, PC-Card, USB
- Vendor innovation and value added features are encompassed and encouraged
- Basic assumption that multiple IFDs can coexist

➔ to be provided by IFD Vendors

The IFD Handler is responsible for exposing the standard interface to the IFD within the PC. It will typically be implemented as part of a device driver, but this is system dependent. The Workgroup specifications are designed to allow flexibility in vendor implementation and functionality may be implemented in software or IFD firmware as desired.

The interface definition is designed to be independent of the I/O interface used by the IFD. It also makes no assumptions about the number of such devices available or the number using a single I/O channel. Possible I/O channels include:

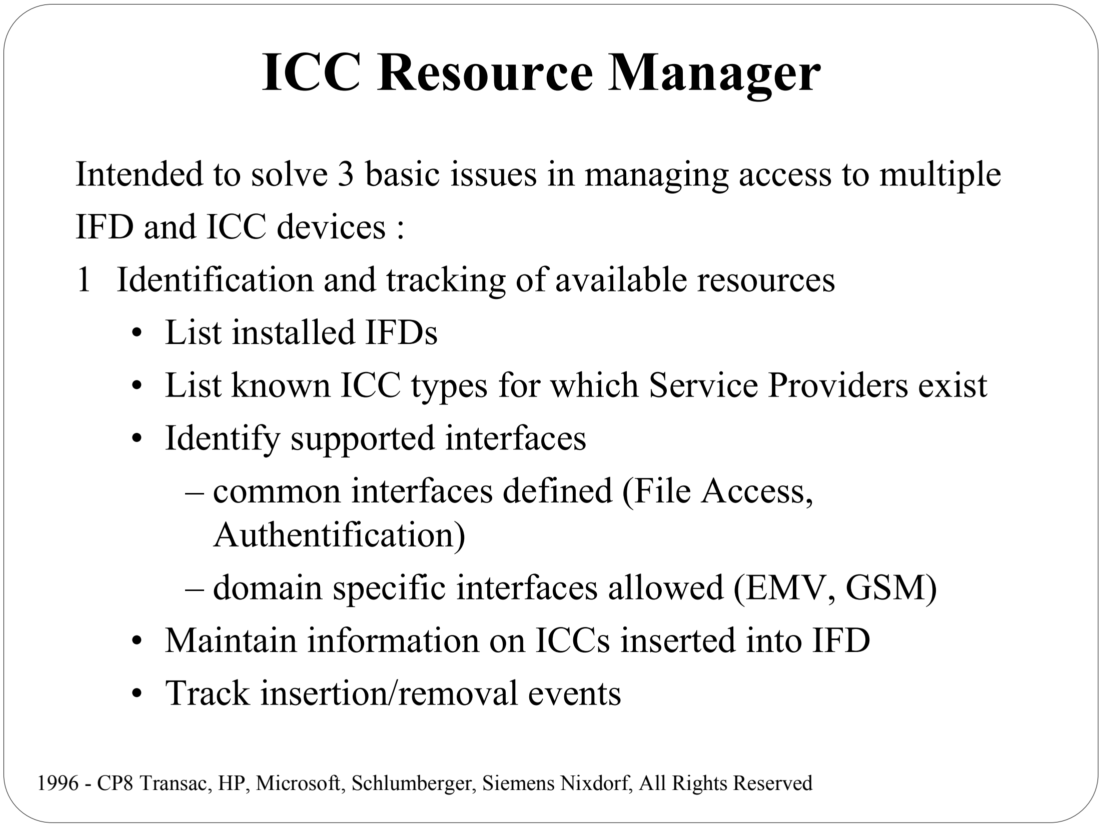
- PS/2 Keyboard Integrated IFD - can be implemented at very low incremental cost beyond basic keyboard. This is likely to be the preferred near term solution for desktop PCs due to the ubiquitous availability of PS/2® keyboard interfaces; the efficient use of desktop space; and the fact such designs do not use other, scarce, I/O resources. It is a convenient location, being situated for easy access by the PC user, and acting as a primary interface to the PC. In particular, it tends to be directly in front of the user making ICC insertion/removal simple and keeping the ICC in view of the user.

- Serial Port (RS-232C) interface - it is possible to build low cost IFDs using standard serial ports. Such devices can easily support multiple data rates, including support for rates in excess of 30 kbps, and are suitable for use with desktop and laptop PCs. Biggest drawback is reliance on serial port that is a scarce resource on most PCs.

- PC CARD based readers - such devices are moderately high cost, but may be the best approach to addressing the needs of existing laptop PCs or sub-PC class devices. Such devices offer growth potential to very high effective data rates, but the design makes ICC insertion/removal somewhat difficult.

- USB interface - efforts are underway to establish USB as a standard for connection of peripheral devices to the PC. It offers a great deal of flexibility, a clean model for handling of multiple devices, and supports data rates up to 10 Mbps. It is believed that USB-based IFDs will eventually supplant PS/2 Keyboard and serial port readers as the preferred designs in the future.

Vendors are free to include value added functionality beyond that defined by these specifications. This could include isolated PIN entry devices or local displays as noted previously, as well as additional functionality. Compliance with these specifications only dictates that these be accessible using the general purpose extension mechanism defined.



ICC Resource Manager

Intended to solve 3 basic issues in managing access to multiple IFD and ICC devices :

1 Identification and tracking of available resources

- List installed IFDs
- List known ICC types for which Service Providers exist
- Identify supported interfaces
  - common interfaces defined (File Access, Authentification)
  - domain specific interfaces allowed (EMV, GSM)
- Maintain information on ICCs inserted into IFD
- Track insertion/removal events

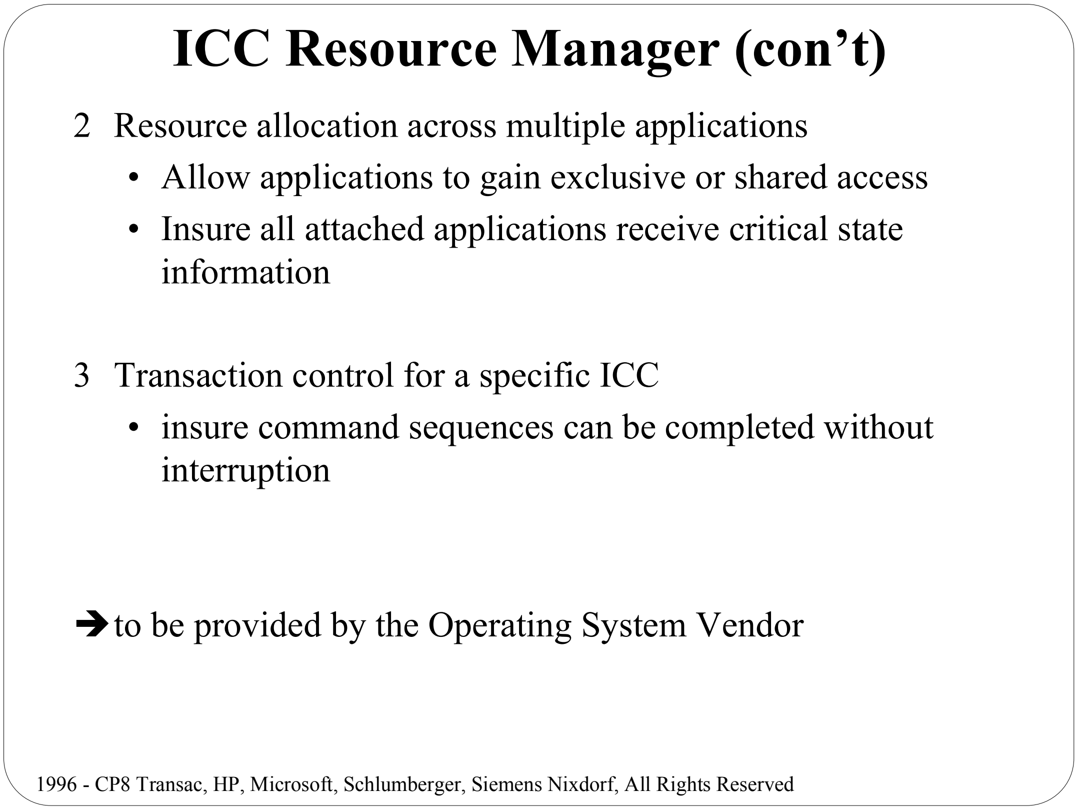
The Resource Manager is a key component of the Workgroup architecture and is designed to address 3 key requirements. It is intended to be implemented as a system service and should be provided by the Operating System vendor.

First, it is responsible for identification and tracking of resources. We recognize that this functionality may overlap with other mechanisms defined within a given system, however, by defining a standard interface, we provide a degree of commonality across systems that simplifies the job of the Service Provider or applications developer.

The Resource Manager should always be able to enumerate the IFDs installed on the system and information about known ICC types. For this latter information it provides a cross reference between ICC type, associated SP/CSP, and supported interfaces. The notion of ICC interfaces is new to these specifications. We believe this will be an important concept moving toward the future when multi-application ICCs and functionality equivalent ICCs from multiple vendors may exist. An Interface is nothing more than a defined collection of services that an application can use. The Workgroup has defined SP "File Access" and "Authentication" Interfaces, since these are widely implemented. Additional Interfaces may be defined to support application domain specific needs.

It is also responsible for tracking actual ICC availability. It does this by acting as a central control point for ICC insertion/removal events. This information will be forwarded to SPs or applications that request it, but the Resource Manager will always track the global state.

One important notion is the mapping of a specific ICC to its associated SP or Interfaces. Since the only information that is guaranteed to be available is the Answer To Reset (ATR) string, we use this information to perform the mapping. There is the potential for conflicts, in which case the user, or application must determine the actual ICC type in use.



ICC Resource Manager (con't)

2 Resource allocation across multiple applications

- Allow applications to gain exclusive or shared access
- Insure all attached applications receive critical state information

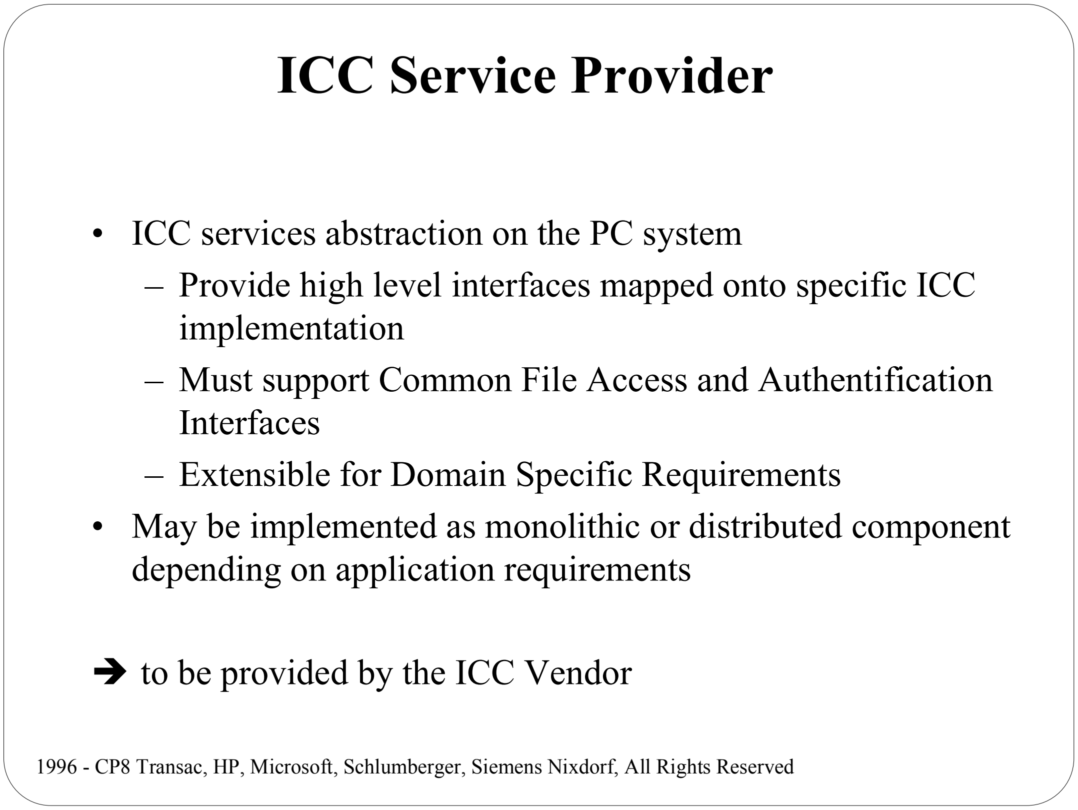
3 Transaction control for a specific ICC

- insure command sequences can be completed without interruption

➔to be provided by the Operating System Vendor

A second function of the Resource Manager is to control device allocation. In a multi-processing environment, there is the possibility that multiple applications will wish to use the same ICC at the same time. The Resource Manager controls access to the ICCs and allows connections to be opened in either shared or exclusive modes.

Finally, it provides transaction primitives. Most operations against ICC require multiple low level commands and may be dependent on preservation of intermediate state information. To insure an application can complete a sequence of operations uninterrupted, it may encapsulate the sequence of command within a transaction. This insures it will have exclusive access for the duration of the transaction.



ICC Service Provider

- ICC services abstraction on the PC system
  - Provide high level interfaces mapped onto specific ICC implementation
  - Must support Common File Access and Authentification Interfaces
  - Extensible for Domain Specific Requirements
- May be implemented as monolithic or distributed component depending on application requirements

➔ to be provided by the ICC Vendor

The ICC Service Provider is intended to provide a high level interface to a given ICC. (It is not intended to expose cryptographic services provided to the ICC, this is discussed on the next slide). The ICC SP definition strives for flexibility. If the ICC exposes file-like entities and authentication services, then it shall expose these using the Workgroup defined interfaces. Note that these services may be implemented in compliance with ISO 7186-4, but need not be.) However, if these services are not provided, then these interfaces need not be supported. The vendor/issuer is also free to provide additional proprietary interfaces required by specific applications.

Another important point is that ICC SPs may be implemented in several different ways. Typically, it is expected they will be implemented as shared libraries on the end-user system. However, this is not the only option. To meet specific application requirements, it may be desirable to implement an SP as a client-server system. This could be done, for example, to incorporate application specific secure messaging operations within a Server security perimeter, while still conforming to this architecture.

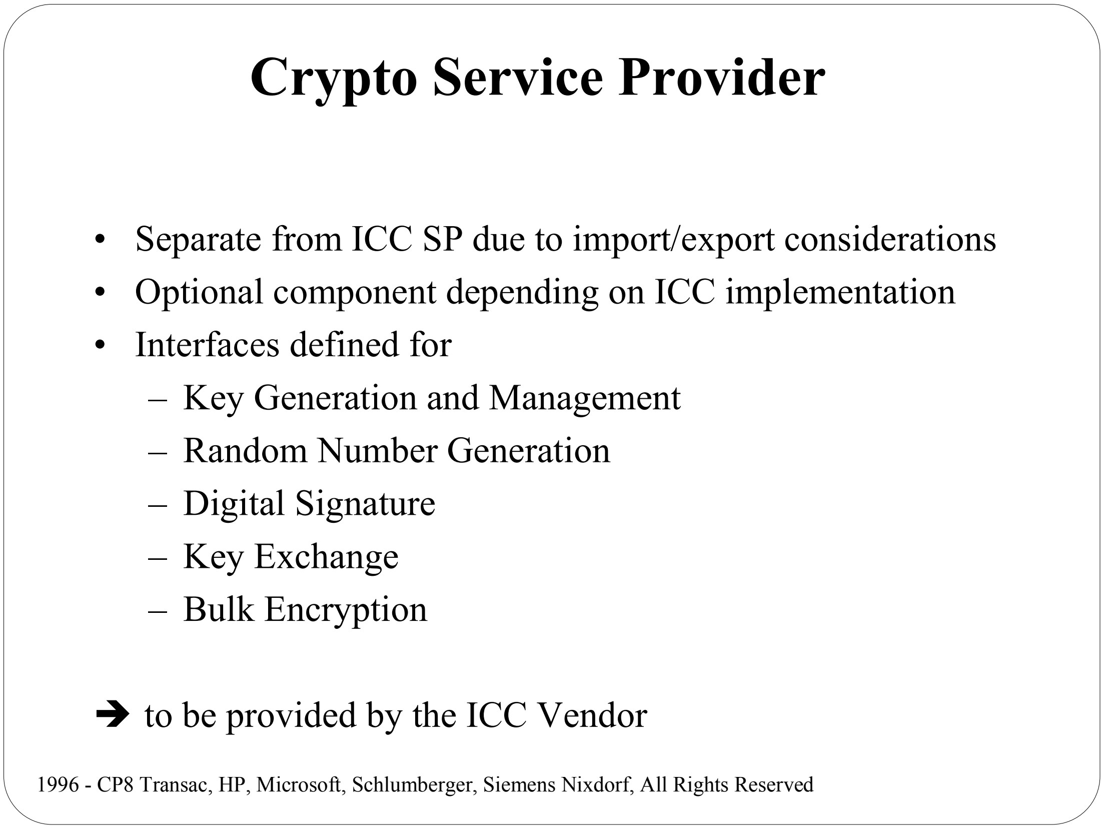
Before an SP may be used within this architecture, it must be "introduced" to the Resource Manager. Typically this is done through a ICC setup utility provided by the card manufacturer. This utility may come with the ICC on a floppy, or may be available on a web site, etc. The setup utility must provide four pieces of information about the card:

1. Its ATR string, and a mask to use to use as an aid in identifying the card.

2. An identifier for the Service Provider(s) which support the ICC

3. A list of ICC Interfaces supported by the card.

4. A 'Friendly Name' for the card, to be used in identifying the card to the user. (In most cases, the user will supply this to the setup utility).



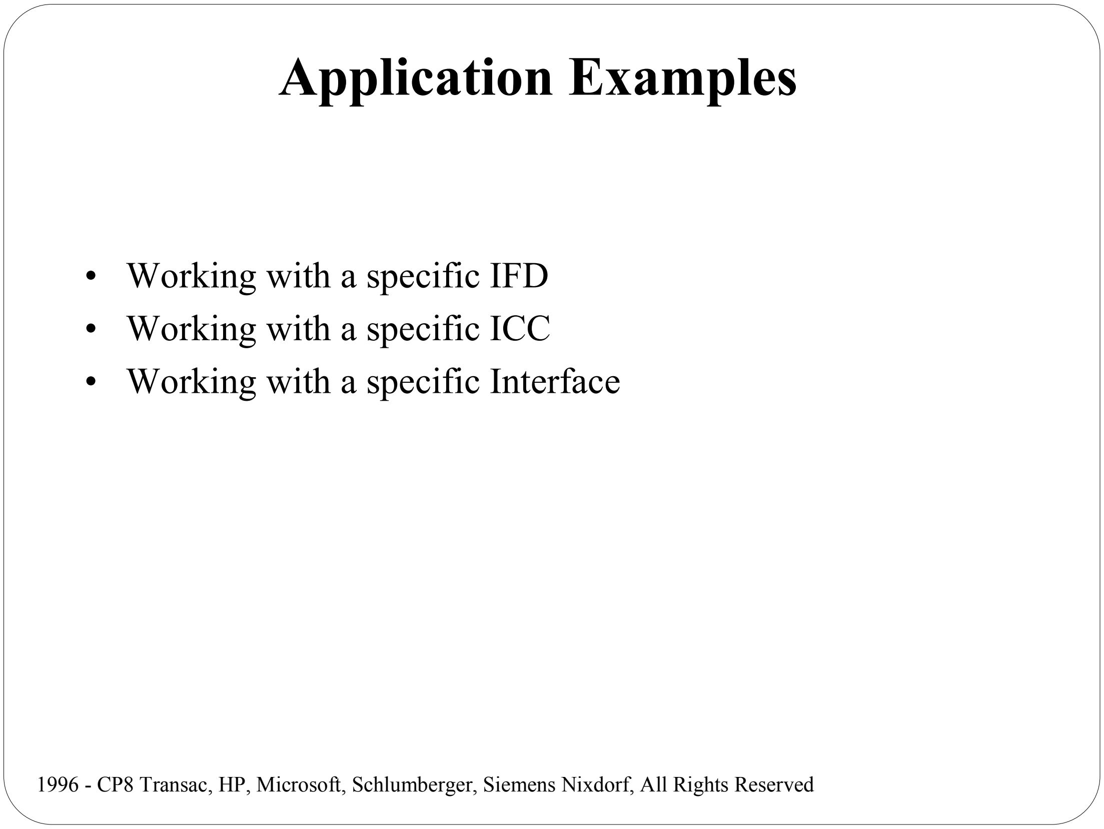
Crypto Service Provider

- Separate from ICC SP due to import/export considerations
- Optional component depending on ICC implementation
- Interfaces defined for
  - Key Generation and Management
  - Random Number Generation
  - Digital Signature
  - Key Exchange
  - Bulk Encryption

➔ to be provided by the ICC Vendor

A distinct Cryptographic SP is defined by the Workgroup architecture. This was done in recognition of the existing issues concerning the export and/or import of cryptographic functionality. The intent is that only ICCs that expose cryptographic functionality usable by external applications would have an associated CSP. Cryptography used internally to the ICC, such as secure messaging, does not require a CSP. To support cryptographic services, Interfaces are defined for making use of the services listed on the slide. Only those services actually supported by the ICC would need to be implemented.

The current interfaces are based on functionality exposed by Microsoft's CryptoAPI, as this is the only currently shipping product that allows for installable Cryptographic Service Providers with well defined export/import properties. Note that this specification does not mandate use of Microsoft's CryptoAPI implementation to expose these services, only that a functionally equivalent implementation be used.



Application Examples

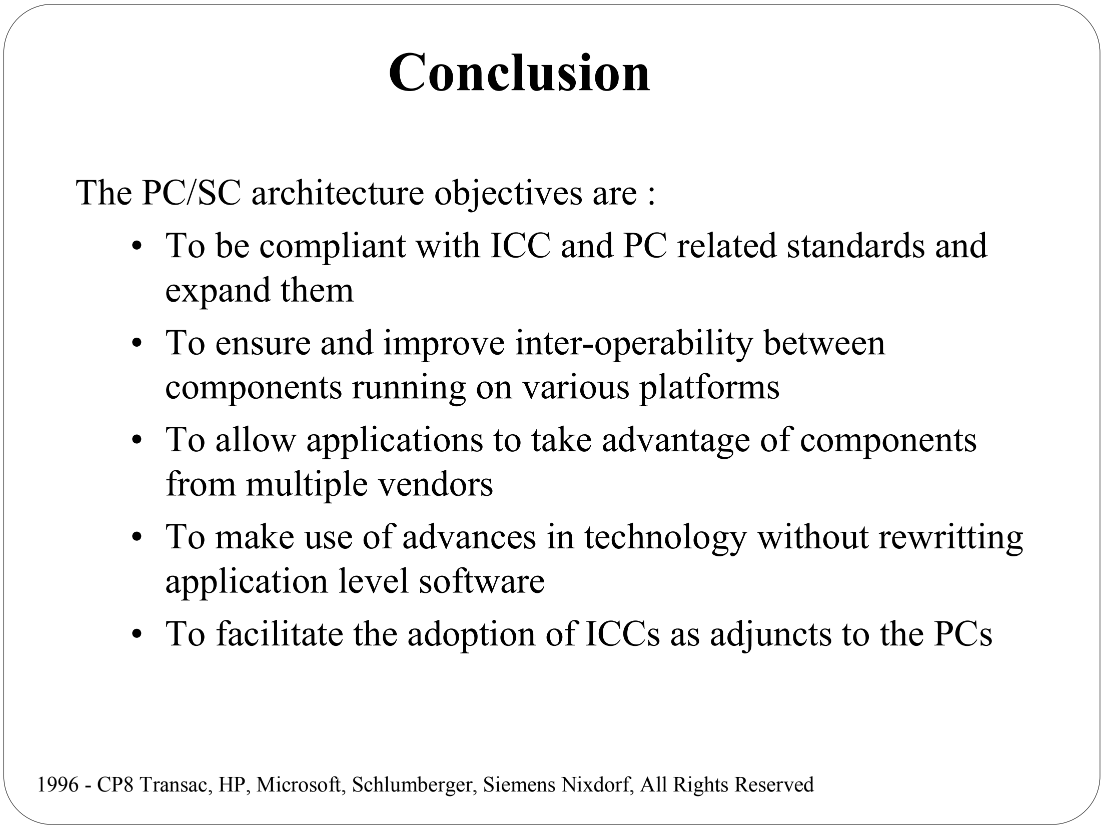
- Working with a specific IFD
- Working with a specific ICC
- Working with a specific Interface

It is important for application developers to understand how the PC/SC Architecture will meet their needs. Hence, it was felt that a brief discussion of how several classes of applications may be supported is in order. These examples are not exhaustive, but hopefully provide insight into how an application developer would make use of the Resource Manager and/or Service Providers.

1. Working with a specific IFD (i.e. Payment System) : Many applications may wish to use a specific IFD. This could be due to its location (i.e., a specific IFD within a public kiosk), or a device with specific functionality (i.e., an IFD that supports a required feature such as local PIN pad). For an application to make use of a specific IFD, it can determine its presence using the Resource Manager, and then monitor that IFD for insertion events. Once an ICC is inserted, the Resource Manager will retrieve the ICC's ATR string and match it against known ICC types. The result is that the application receives a list of possible ICC types and associated SPs which could support that ICC (unless the application knows this a priori). The application can then connect to the ICC using the SP to provide appropriate high level interfaces. Note that this model allows an application to monitor a specific reader and then easily spawn a process which deals with a specific type of ICC/application.

2. Working with a specific ICC (i.e. Personalization Station): This type of application will likely be designed to work with a specific ICC . Such an application could use the Resource Manager to check for the required SP(s) and IFD if necessary. However, in normal operation, one would expect the application to simply connect to the proper SP and wait for an ICC insertion event. It would then run through a series of operations against the ICC, release the ICC, and wait for the next insertion event.

3. Working with a specific Interface (i.e. Web related applications) : In this instance, an application requires access to information storage and retrieval capabilities. It doesn’t care which IFD an appropriate ICC is inserted into, and may rely on the user to select a specific ICC to use. In this case, the application can simply wait for the first ICC inserted into any IFD (or subset of IFDs) which emits an ATR associated with an ICC type supporting the desired interface. At that point, the application may connect to the ICC and make use of the available services through the proper SP(s).



Conclusion

The PC/SC architecture objectives are :

- To be compliant with ICC and PC related standards and expand them
- To ensure and improve inter-operability between components running on various platforms
- To allow applications to take advantage of components from multiple vendors
- To make use of advances in technology without rewritting application level software
- To facilitate the adoption of ICCs as adjuncts to the PCs

1996 - CP8 Transac, HP, Microsoft, Schlumberger, Siemens Nixdorf, All Rights Reserved

The PC/SC architecture proposes to reach the following goals :

- be compliant with existing ICC and PC related standards and expand them where necessary;
- ensure and improve inter-operability between components running on various platforms. Specification is designed to be implementable on a variety of hardware and software platforms, such as UNIX, Windows, Macintosh, NC, ….
- allow applications to take advantage of products and components from multiple manufacturers (vendor-neutral),
- make use of advances in technology without re-writing application level software. (application-neutral),
- create standards for application level interfaces to ICC services in order to facilitate adoption for broad range of PC applications,
- create an environment which encourages the adoption of ICCs as adjuncts to the PC.

Every member of the Workgroup is committed and strongly motivated to promote the architecture.