# Interoperability – The Truth

**John Lees**
Transmax Pty Ltd, 37 Redgum Place, Gaythorne, Queensland 4051 Australia
Phone: +61 7 3355 8700, Fax: +61 7 3355 8799, Email: John.Lees@transmax.com.au

## ABSTRACT

This paper compares interoperable and integrated system architectures for significant applications within an organisation. These architectural styles are defined and then compared using the following criteria: computing resource requirements, configuration data management, interface management, systems testing and user interfaces. The architectural styles used in other more mature disciplines are then reviewed. The *best of breed* advantage of interoperable architectures is discussed and the STREAMS integrated intelligent transport system architecture that incorporates this capability is described. The paper concludes that an integrated system architecture results in lower costs, better performance and lower risks than a comparable set of interoperable systems for significant applications within an organisation.

## KEYWORDS

Integration, Interoperability, System Architectures

## INTRODUCTION

Interoperability between associated intelligent transport systems (ITS) is essential and is widely supported by standards bodies, suppliers and users. Interoperability is fundamental to modern computing and is necessary for communications between loosely related systems and between different organisations. However, interoperability is not a panacea and is often not the best approach for major ITS applications within an organisation.

Sets of closely related interoperable computer systems within an organisation can be difficult to manage and often fail to achieve their objectives. Those who do not understand the potential difficulties or who have not experienced their problems often promote extensive interoperable architectures within organisations. However, the awful truth is that a set of related interoperable systems within an organisation can be both relatively expensive and difficult to sustain.

## SYSTEM ARCHITECTURES

Computer system architectures can generally be classified as either interoperable or integrated. Most modern computer installations include some mix of both these architectural styles and the classification of a particular system tends to depend on the degree of integration / interoperability.

### Interoperable System Architectures

Interoperability occurs where computer systems exchange information with other systems. Interoperable architectures partition a set of closely related functions into separate computer systems that exchange information with each other to achieve the required business outcome. Systems within an interoperable set typically perform a subset of the functions required for the business outcome and exchange information with related systems via external data communications interfaces.

### Integrated System Architectures

Integration exists where computer systems aggregate larger numbers of related functions within a single system. Integrated architectures typically partition a set of closely related functions into subsystems that exchange information with other subsystems through software interfaces and internal data communications interfaces. Integrated architectures seek to minimise data communications interfaces with external systems. They combine a larger number of related functions and have fewer data communications interfaces to external systems.

## INTEROPERABLE AND INTEGRATED SYSTEMS COMPARED

Interoperable systems and integrated systems are compared in the following subsections:

1. Computing resource requirements
2. Configuration data management
3. Interface management and system testing responsibilities
4. User interfaces.

### Computing Resource Requirements

The applications software and systems software have become closely coupled in modern computer systems. The applications software performs the business function and is now often dependent on specific versions of the operating system, database management system, geographic information system and the data communications software. In many cases, this coupling between the applications software and the systems software is such that it is not possible to run more than one significant application on the same hardware platform.

Where this situation occurs, separate host processors, user workstations and field processors can be required for each major application. The increase in computing resource requirements can be significant. For example, the traffic signal management system, the motorway management systems and the passenger information system might require separate workstations. This has implications for both desk space and office space.

A number of strategies are being used to minimise this problem and, in particular, reduce the need for multiple user workstations. They include:

1) Standard operating environments. Many large organisations specify a standard operating environment and mandate that all user workstation applications software runs within this environment. This seems good in theory but can have difficulties in practice. These include:

   a) Deployment of different versions of the standard operating environment. In large, geographically diverse organisations, it can take a long time to deploy the same version of the environment. Where this occurs, it affects the deployment of the application systems and concurrent operation of different versions of the application software can be inefficient.

   b) Standard operating environment inertia. Major versions of these environments often tend to remain in place for long periods. This can prevent the implementation of new system software functionality to the detriment of the business. This could be significant in this age of data network security threats.

2) Thin client applications. Thin clients (Windows Terminal Services, Citrix, etc.) can overcome the need for separate workstations, but this often comes at the cost of reduced speed and reduced functionality. Users are often not willing to pay the price.

3) Platform independence. Java is the classic example of this approach but to date it has not always been the expected panacea. Performance can suffer and platform independence is not always achieved. Hence the anti Java comment - *write once, debug everywhere*.

It is concluded for the reasons outlined above that a set of closely related interoperable systems within an organisation currently requires more computing resources than an integrated system performing the same functions. These computing resources include: hardware, systems software, application software and data communications. These additional resources usually increase total system complexity and increase both the capital costs and the operating costs.

## Configuration Data Management

The configuration data for many computer systems can be classified into data describing the system's operating environment and data specific to the business function. With intelligent transport systems, the environment configuration data can be further classified into data related to supporting subsystems and data relating to the field equipment. The configuration data for intelligent transport systems can therefore be classified as follows.

1) Supporting subsystems data. This includes data describing the following:

   a) The transport network being managed. In modern systems, this data is usually geographic data because most users prefer map-based systems.

   b) The distributed computer system. The host processors, field processors, workstations and the associated data communications network.

   c) User access control. An extension of the operating system's access control functionality can be required where multiple agencies own or operate the field equipment and delegation of authority is required.

   d) Event and alarm reporting requirements.

2) Field equipment data. This includes the configuration data associated with vehicle detectors, dynamic message signs, traffic signal controllers and environmental sensors.

3) Intelligent transport subsystems data. This includes the data specific to the intelligent transport function. These include:

   a) Traffic signal management algorithms

   b) Motorway management algorithms

   c) The incident management subsystem

   d) The driver and passenger information subsystems.

Much of this supporting and field equipment configuration data must be specified for each system where a set of separate interoperable systems perform the traffic signal management, motorway management and other functions within the same region. This usually increases the organisation's implementation and maintenance costs and this can be particularly significant where geographic data is involved. Overall performance can be compromised because of the inconsistencies that inevitably occur when the same item of data exists within multiple systems.

However, where the traffic signal management, motorway management, incident management and other functions are integrated within a single system, the supporting and field equipment configuration data only needs to be specified once. This reduces the installation and maintenance costs and improves performance because inconsistencies are reduced. Figure 1 illustrates this principle.

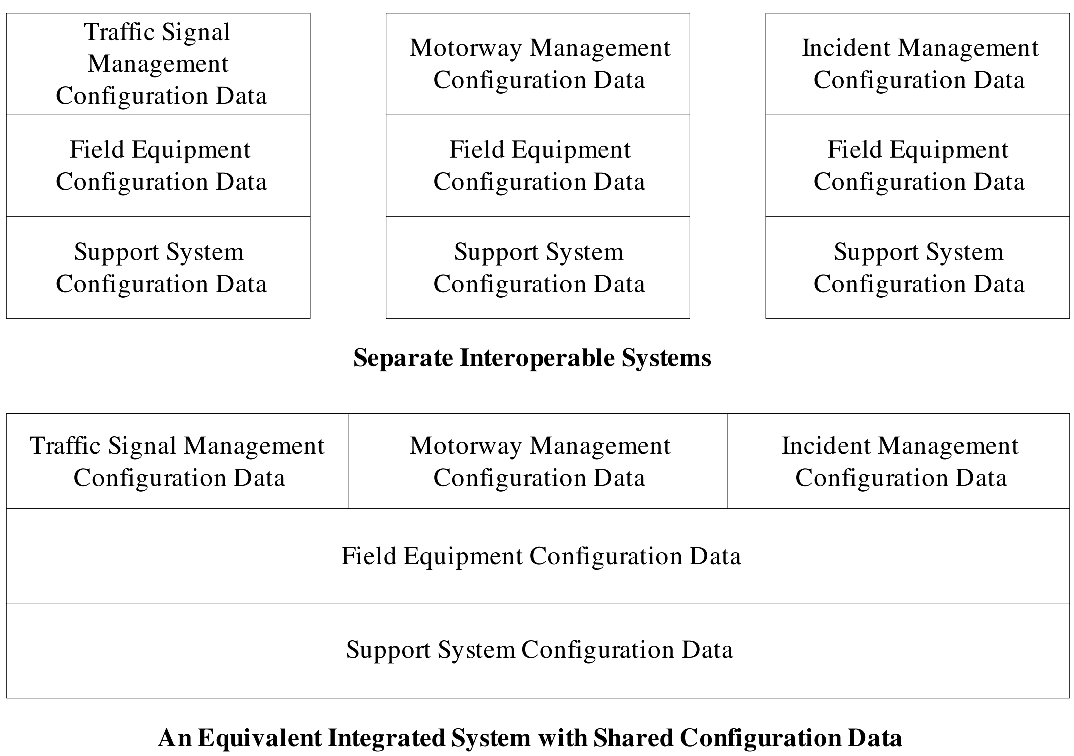


**Figure 1 Interoperable vs. Integrated Systems**

Intelligent transport systems have been used in this example; however, the same principles apply in other disciplines. A set of interoperable systems generally requires more data specification and maintenance than a comparable integrated system. This increases the data management costs and increases the potential to compromise performance through inconsistencies.

## Interface Management and System Testing Responsibilities

Interface management and system testing is a significant task with distributed process control systems such as intelligent transport systems. The system owner or system integrator usually has responsibilities for these tasks where a set of interoperable systems from different suppliers are used within an organisation. The system owner or their agent must specify and test the required data communications interfaces between the systems within the interoperable set. This involves the system owner or agent in specification, negotiation, vendor management and system interoperability testing tasks throughout the lives of each of the interoperable systems. This is ongoing because these processes must be repeated with each new version of any of the interoperable systems in the set. This increases the level of operational management for the system owner and therefore the associated operational risk.

Many would argue that this should not be a problem if the relevant data communications interface protocols were properly specified, implemented and tested by each of the interoperable systems. However, the specification, implementation and testing of interface protocols can be imperfect and problems occur when this happens.

The author recently experienced a situation where significant traffic operations problems occurred for 12 days after one of two interoperable systems was upgraded. The data communications protocol between the systems was not changed and had been operating without problems for many years. The supplier of the system that was unchanged argued for 10 days that the bug must have been in the system that had been changed. Not surprisingly, the system owner agreed. The bug was eventually found to be in the system that hadn't been changed. The bug had existed for many years and was triggered by a slight change in the order of the messages between the systems. (Modern computer systems are usually *event driven*. Therefore, the order of messages within the data communications interface protocol cannot normally be defined.)

With a comparable integrated system, the interface management and testing responsibilities transfer to the system supplier. Similar problems to those described above can occur but their resolution is the supplier's responsibility and they should be resolved during the supplier's integration and system testing processes. If bugs are missed and occur during operations the integrated system supplier is responsible. There can be no *pointing the finger at the other guy*.

Interface management and system testing with interoperable systems are the responsibility of the system owner or their agent and they bear the associated risks. With integrated systems performing comparable functions, these tasks are the responsibility of the system's supplier and the associated risk transfers from the system owner.

### User Interfaces

Traffic management centres typically perform multiple functions – traffic signal management, motorway management, incident management, etc. When these functions are performed by separate, interoperable systems, the traffic management centre operators need to be familiar with multiple user interfaces. Similar tasks may need to be performed differently in different systems and terminology differences sometimes occur. This increases operator training costs and can increases the likelihood of operator error and consequential problems.

Multiple user interfaces can be replaced by a single, consistent user interface when the same set of functions is performed by a single, integrated system. This should decrease operator training costs and decrease the likelihood of operator errors.

## THE SITUATION IN OTHER DISCIPLINES

The debate between interoperability and integration has been decided in favour of integration in many other disciplines. For example, the road design function is now normally undertaken using an integrated system that encompasses survey, 3-D coordinate geometry, hydraulics, earthworks, pavement modelling, etc. Previously these functions were each undertaken by separate systems with varying degrees of interoperability.

Business and government now perform their financial management functions using integrated systems such as SAP, Oracle Financials, MIMS, etc. These integrated systems have replaced the separate packages that previously managed purchasing, accounts payable, accounts receivable, payroll, inventory, etc.

The older approach of sets of systems each performing one particular function is still common with intelligent transport systems because it is a relatively new and comparatively immature industry. In the author's experience, the ITS installations with separate systems have often not achieved the interoperability originally proposed.

## THE BEST OF BOTH WORLDS

### Best of Breed

Interoperable systems have the advantage of being able to select the *best of breed* for each function. This means that each system could be selected as the best available for that particular function. It is argued that the best overall outcome could then be achieved by combining a number of best of breed systems into an interoperable set. For example, the best traffic signals management system, the best motorway management system and the best incident management system.

While this does seem an attractive approach, experience has shown that it often does not produce a good overall outcome. However, this ideal could become a reality if the disadvantages of interoperable systems described earlier could be overcome.

At the interoperable extreme, the system complexity, cost and difficulty implementing and maintaining the interfaces often means that the desired business outcomes are never achieved. At the integration extreme, the system's size can also produce unmanageable complexity and the same negative outcome can occur.

## The STREAMS Architecture

The STREAMS enterprise intelligent transport system has been built using an integrated architecture. However, it also includes best of breed capabilities. The architecture classifies the software within the system into the three major categories described above in the configuration data subsection. They are:

1) Support subsystems. These include:
    a) The transport network management subsystem (GIS).
    b) The distributed computer system management subsystem.
    c) The publish / subscribe subsystem.
    d) The user access control subsystem.
    e) The event and alarm reporting subsystem.
2) Field equipment subsystems. These include:
    a) The vehicle detector subsystem.
    b) The dynamic message sign subsystem.
    c) The vehicle location and identification subsystem.
    d) The traffic signal controller subsystem.
    e) The ramp meter subsystem.
    f) The environmental sensors subsystem.
3) Intelligent transport subsystems. These include:
    a) The traffic signal management subsystem (the algorithms).
    b) The motorway management subsystem.
    c) The incident management subsystem.
    d) The passenger information subsystem.
    e) The driver information subsystems (including parking guidance).

The intelligent transport subsystems (eg motorway management) receive measurements from the relevant field sensor subsystems and implement their requirements using the appropriate field actuator subsystems. The support subsystems serve all other subsystems. All software associated with each particular function is encapsulated within one subsystem and all other subsystems requiring a particular service ask the relevant subsystem to perform the function.

The system has been designed using the *Lego Block* principle so that individual subsystems can be replaced with an improved subsystem with minimum impact on other subsystems. This allows the system to be upgraded into the future without redevelopment of the entire system. It also enables *best of breed* functionality to be incorporated for particular services without change to unrelated functions.

A diagram describing the STREAMS architecture follows (Figure 2):

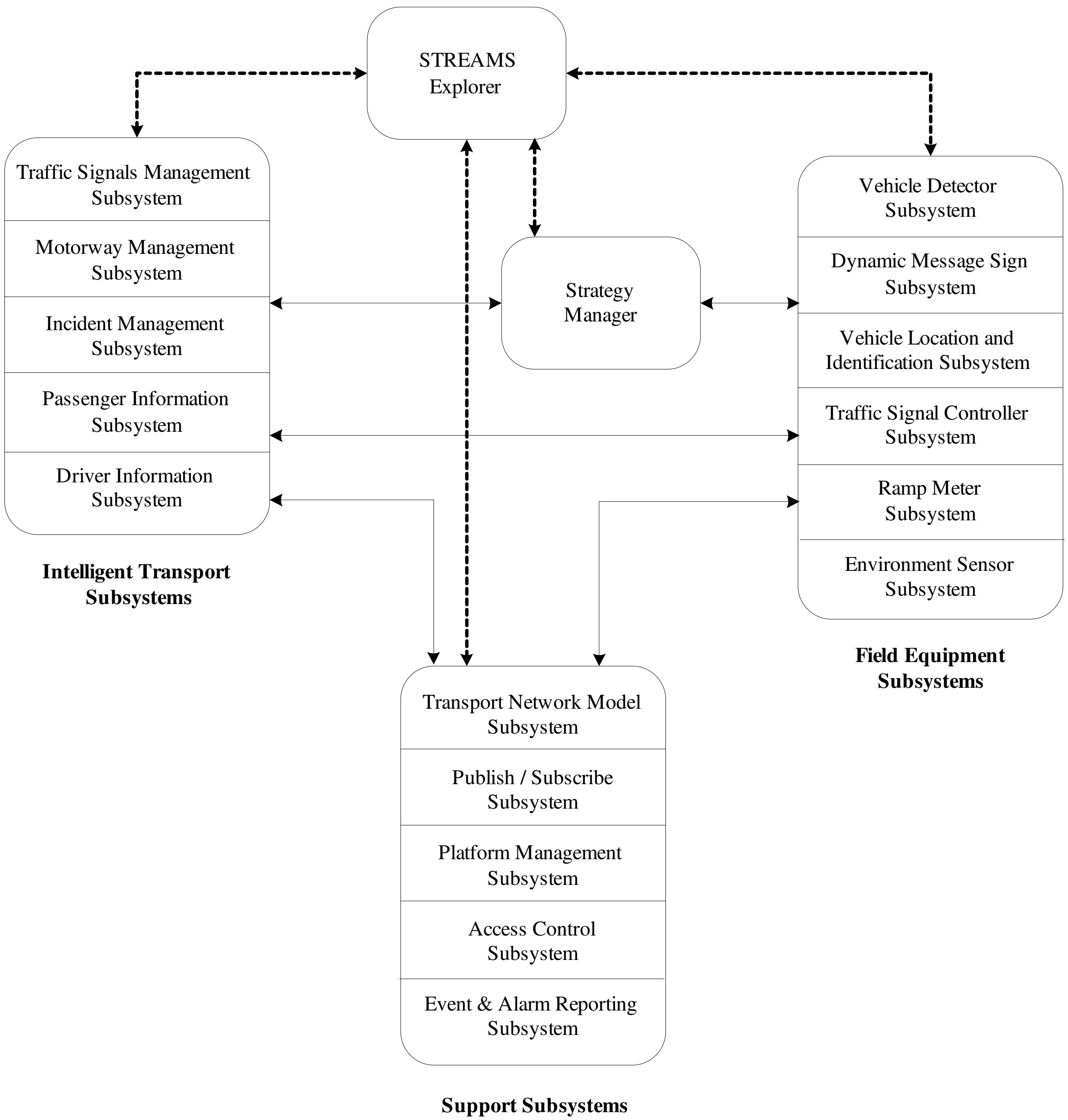


**Figure 2 STREAMS – An Integrated System with Replaceable Subsystems**

## CONCLUSION

A set of interoperable systems performing a significant application within an organisation has a number of disadvantages when compared with an integrated system performing the same functions. These include:

1. They require more computing resources and this increases capital costs and operating costs.
2. Duplication of the configuration data is required and this can increase errors, increase risk and reduce performance.
3. Interface management and total system testing responsibilities transfer from the supplier to the system owner or their agent.
4. The user interfaces usually vary and this increases training costs and the potential for errors.

The interoperability vs. integration debate has been decided in favour of integration in most disciplines. It is only now being considered within the ITS industry because of its relative immaturity. However, large integrated systems can have significant inertia and this makes them inflexible and difficult to upgrade.

An architecture that seeks to combine the advantages of integration with the ability to upgrade modules with minimum change to other subsystems is considered to incorporate the best features of both architectural styles. The enterprise intelligent transport system STREAMS exemplifies this architecture.