

**wwiteware™**  
**Service Delivery Platform**  
**A Directory Service Evolution**

V1.5 – October 2010



## Table of Contents

<b>1</b>	<b>Introduction .....</b>	<b>4</b>
1.1	<i>Online systems have evolved.....</i>	4
1.2	<i>The age of information engines and online customers: introducing wwiteware .....</i>	4
1.3	<i>What wwiteware brings to the table .....</i>	4
1.4	<i>Departure from tradition.....</i>	5
<b>2</b>	<b>IT Architectures.....</b>	<b>5</b>
2.1	<i>IT architectures - are new tools of the trade required?.....</i>	5
2.2	<i>System Engineering Paradigms .....</i>	6
2.3	<i>Databases and Directories are Different .....</i>	6
2.4	<i>The wwiteware Service Delivery Platform.....</i>	6
2.5	<i>Telco and MSO SDPs.....</i>	8
2.5.1	<i>Telco SDPs – Mobile Services.....</i>	8
2.5.2	<i>SDFs Are Vague.....</i>	8
2.5.3	<i>MSO SDPs – Information Engineering .....</i>	9
2.5.4	<i>Telco SDPs Evolve to MSO SDPs .....</i>	9
<b>3</b>	<b>Databases and Directories.....</b>	<b>10</b>
3.1	<i>Databases.....</i>	10
3.2	<i>Directories .....</i>	10
3.3	<i>IT Systems.....</i>	10
3.4	<i>Data Models.....</i>	11
3.5	<i>Conclusion .....</i>	11
<b>4</b>	<b>Information Engineering .....</b>	<b>11</b>
4.1	<i>Directories - the identity design of front of house systems .....</i>	11
4.1.1	<i>Identity engineering .....</i>	12
4.1.2	<i>New generation system information taxonomy.....</i>	12
4.1.3	<i>Information agility .....</i>	13
4.1.4	<i>Information scale and demands .....</i>	13
4.1.5	<i>User entitlements .....</i>	13
4.1.6	<i>Click through management .....</i>	14
<b>5</b>	<b>Transactions and Managed Information.....</b>	<b>14</b>
5.1	<i>The Classic IDM-Directory Architecture.....</i>	14
5.2	<i>Adding Information and Management .....</i>	15
5.3	<i>The Front and Back Office .....</i>	15
5.4	<i>The SDP – “The IDM-IAM Evolution” .....</i>	16
5.4.1	<i>SDP Functionality within an Architecture .....</i>	16
5.4.2	<i>SDP Functionality and its Underlying Infrastructure .....</i>	17
5.4.3	<i>Operational Requirements .....</i>	17
5.4.4	<i>Service Assignment – Infrastructure Identity .....</i>	18
<b>6</b>	<b>Data Models, Objects and Their Management .....</b>	<b>18</b>

6.1	<i>Data models require management</i>	18
6.2	<i>Directory Object Management</i>	19
6.3	<i>wwiteware Directory Object Management</i>	20
6.4	<i>The self-care real time online world</i>	21
<b>7</b>	<b>The wwiteware revolution</b>	<b>21</b>
7.1	<i>The Front Office</i>	21
7.1.1	Group Identity Management	22
7.1.2	Information Environment	22
7.1.3	Presence and Events	22
7.1.4	Directory Service Utility	22
7.1.5	Directory Information Scope	22
7.1.6	Entitlements and Security	22
7.1.7	Devices and Infrastructure	22
7.1.8	Information Performance	23
7.1.9	User and Systems Management	23
7.2	<i>Group Identity Management</i>	23
7.3	<i>The Information Environment</i>	24
7.4	<i>Presence and Events</i>	24
7.5	<i>Directory Service Utility</i>	25
7.6	<i>Information Scope</i>	25
7.7	<i>Entitlements and Security</i>	26
7.8	<i>Devices and Infrastructure</i>	27
7.9	<i>Information Performance</i>	28
7.10	<i>User and Systems Management</i>	29
<b>8</b>	<b>Conclusion</b>	<b>30</b>
	<b>Appendix 1 - The Application of LDAP</b>	<b>30</b>
	<i>Current LDAP Status:</i>	30
	<i>LDAP Issue: Micro-Mechanisms</i>	31

# 1 Introduction

## 1.1 Online systems have evolved

The last 30 years have seen tremendous change in the use and growth of online systems. Traditional systems comprise mainframes and back office application databases that perform user transactions on dedicated processes.

Today we talk of identity management, web services, clouds, social networks, customer centric systems, self care and mobile online service environments.

More importantly as our online systems evolve and the demands on them increase, we must continually examine if we can use the same architecting methods of interconnect, functional processing and data management of the past without risk, cost overruns or even IT project failure. For example, if an online system has 10 million human users (all with personalised profiles) and 10 million end user devices of various types, all interacting in real time and affecting the operator's costs and revenues, how is this critical functionality represented in our current transactional architectures or high level modeling methods?

This paper discusses the evolution of thinking about directory services. To be provocative, at this point in time in the IT industry, we really do need to provoke discussion. Therefore we are challenging the historical approaches to the architecting of systems – they do not deliver to customer expectations, nor do they deliver the needs of commercial operators of low cost and new, flexible customer centric online strategies.

And we propose wwiteware and the engineering approaches and discipline it encompasses as a critical challenge to the data base and directory providers that do not offer a platform for business and service growth.

The paper is designed for IT architects. We explain the evolution of our thinking in information design and online services delivery in order to better explain wwiteware.

## 1.2 The age of information engines and online customers: introducing wwiteware

Companies around the world are now using powerful internet search engines to respond to the real time nature of the web and the expectations of its users: as a result a completely different set of design rules is required.

wwiteware follows this evolution toward functionally-rich information processing engines. wwiteware is a Service Delivery Platform (SDP), an engine for that end of the online business where customers are serviced and value is delivered and companies gain commercial benefit.

wwiteware takes a new approach to information processing and uses a highly evolved intelligent directory repository. wwiteware's unique approach to design enables it to address emerging information system and identity processing demands that traditional process-centric architectures are now struggling to deal with.

wwiteware takes the design focus away from the traditional "data centric" world with its focus on back-office processing, to the online event-driven information world putting the needs of the online customer first.

## 1.3 What wwiteware brings to the table

Let's take an example: imagine 20 different process-centric IT functions, typically used by your online customers today, where each function runs on a different server, using its own dedicated data model, identity regimes, management systems and reporting tools. Let's estimate that such a system uses 40 million lines of software with different version control issues and with 25% of that code doing nothing more than moving and translating data

between the fragmented IT functions. Operationally such systems struggle to keep up with the load.

Now imagine what it would be like if we could take that 40 million lines of code and reduce it by around 90% through the application of coherent identity and information management doctrines and advanced directory services technology, whilst adding more functionality. And lastly, imagine being able to demonstrate such a system prior to its operational deployment in a few months rather than years.

This is what we have done with wwiteware.

## **1.4 Departure from tradition**

wwiteware departs from and even contradicts some of the traditional approaches to directory system and service delivery architectures.

For example, in appendix 1, we describe the Lightweight Directory Access Protocol (LDAP) server and the LDAP standards (Request For Comment - RFCs) all of which are normally requested when directory service technology is considered within an architecture.

We conclude in that appendix:

“The wwiteware agenda is to simplify the complete system design , its implementation and its cost by providing managed service based technology which can be demonstrated immediately, easily applied to application development and evolved into operations.

In contrast, the LDAP server and RFC approach is literally starting a large scale, customer centric, business system and application design from a data repository protocol, possibly even down to the single data attribute level... That is: starting the whole design from “ground zero”.

The time, cost and performance risks involved in the LDAP repository approach are all unnecessary risks for management to take.

We now describe the differences and discuss the key thinking behind the wwiteware approach.

wwiteware is in its early stages of release, it is demonstrable and in the project ready state. wwiteware is being marketed in a number of countries.

## **2 IT Architectures**

### **2.1 IT architectures - are new tools of the trade required?**

IT architecture has fallen into the dangerous habit of ‘connectivity’. We challenge that and propose an information engineering approach. Why? Because it begins with an understanding of the value that must be delivered by the system to those that will manage it, use it and pay for it.

There are many types of architectures applied to IT systems. IT architectures can be represented by high level models or abstracted elements, the hardware and cabling components, the networking and functional components, the database data model and application design, the information and management schemas and so on.

In designing new systems, it is often the case that abstracted element diagrams or networking and functional architectures exclude much of the information, identity, management and operational engineering issues which then have to be solved as the project evolves – at a cost.

Customer-centric systems need to be scalable, robust and cost-effective for a business, new “tools of the trade” are fundamental.

## **2.2 System Engineering Paradigms**

Online system architectures have two data repository paradigms and two operational paradigms:

For the data repository paradigm, we know that:

- relational databases are well suited to the transactional processes that require data models that are dedicated to a particular application, and
- directories contain identified information objects that in simple cases relate to the human user's access to a system. But in more sophisticated situations the directory information objects relate to the identified resources of an online system that engage in the management and consumption of system services.

For the operational paradigm:

- there is the back office where transactional applications and databases reside, and
- there is the front office where the user access and self care systems reside.

Straddling the boundary between front and back are the session control functions and the product and services management functions - both of which are critical.

Typically system designs place process-oriented repositories at the back of a system and the identified object repositories at the front of a system. With this “information” engineering doctrine we recognize that users can search (for information, etc) a business at different rates to those users who may actually authorise themselves to do transactions with the back office.

## **2.3 Databases and Directories are Different**

Existing databases and directories are seen as “data” storage and retrieval functions. Both have evolved over the years and are very different in terms of their data modeling approaches, information engineering and access and management methods. Relational database (RDB) technology was developed in the 1970's and uses SQL to access the data.

Directories came into being with the telephone system white pages services in the early 1900s but it was not until the 1980's that they emerged in the IT world, with networked operating systems such as Novell and Banyan and in the standards arena, X.500 followed by the LDAP – X.500 access protocol design.

Other commonly used directory services include the Home Subscriber Store (HSS) and Home Location Register (HLR) of the mobile phone world and of course the internet's global Domain Name Services (DNS).

Both databases and directories are fundamental to all systems today. They are not going away in the foreseeable future, however they will evolve.

## **2.4 The wware Service Delivery Platform**

The wware technology development takes a view that existing LDAP directory service technologies must evolve to be faster, distributed and to provide much better utility from an identity management, user service provisioning, presence and online user experience perspective. This directory service evolution will also determine how other parts of a system, such as directory enabled applications and subsystems, are designed and deployed.

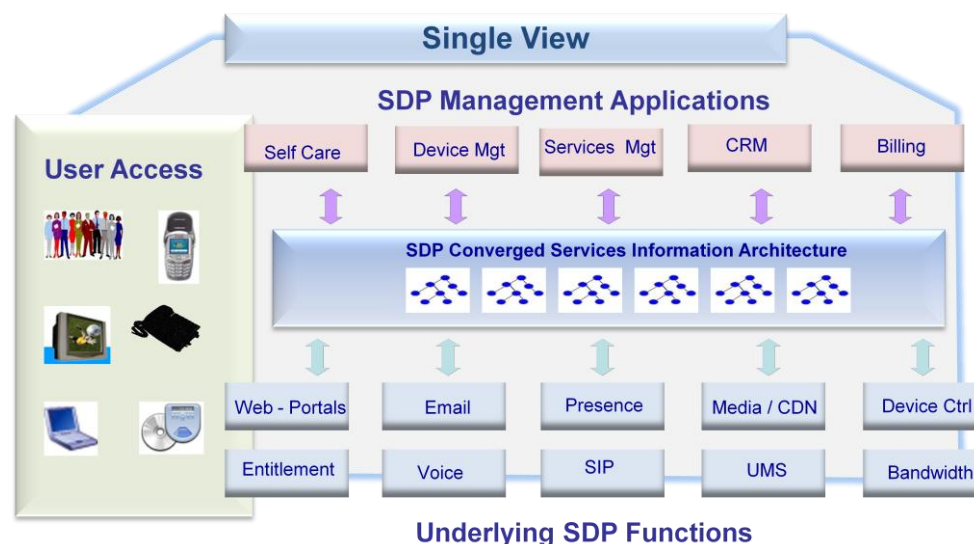
The wwiteware technology addresses the requirement that IT systems must be designed with much more functionality than “a repository”, “an application”, servers and some “interfaces”. To this end, wwiteware is a customer-centric directory engine with a range of system management functions that form a Service Delivery Platform (SDP).

**A SDP is a grouping of functions that provide a coherent integration platform on which services are provisioned, controlled, delivered and measured to meet the customer’s personalised service requirements.**

SDP’s provide critical functionality for an online business because they manage how:

- A customer-centric, company-wide information model is used for the online business.
- Customers access their products, services and self-care functions.
- Product and services managers and operational support staff administer the online service delivery systems.
- The back office applications and management systems (such as CRM, service activation, billing and other OSS systems) relate to the front of house operations.
- The underlying IT and communications infrastructure functions are controlled from a services delivery perspective.

The diagram below identifies how the SDP applies itself to an online system and highlights its key management functions such as “single view”.



**The critical points are:**

- Customer centricity generally includes how the online user experience is managed and should also include from a business perspective how (online) customer acquisition is orchestrated and how third party services are enabled.
- Today large-scale directory systems are being developed for customer centric operations, the on-line user environment, security, end user service provisioning and identity management functions.
- With the need to contend with increase in personalization, complex authorization rules, location and device type issues, the system is much simpler when the authentication and authorization service is performed with one directory access (instead of dozens of accesses) using the advanced processing features of the wwiteware directory.
- Advanced directory services should have a capability that allows the critical parts of the design to be adapted and incorporated into integrated circuits (silicon). wwiteware accommodates this potential.

## 2.5 Telco and MSO SDPs

Alan Lloyd of wwrite has been involved with identity engineered Multiple Service Operator (MSO) style SDPs since the mid 1990s primarily for defence systems, large organisations, ISPs and converged service operators.

In 2001-2002, there was a need for the telco world to enhance its service management around mobile phone platforms in order to deliver content, presence and push to talk-type services i.e. service creation capabilities around telephone communications. The telco industry groups and their vendors refer to their technologies as SDPs. However, the two styles of SDP (MSO and telco) are quite distinct.

### 2.5.1 Telco SDPs – Mobile Services

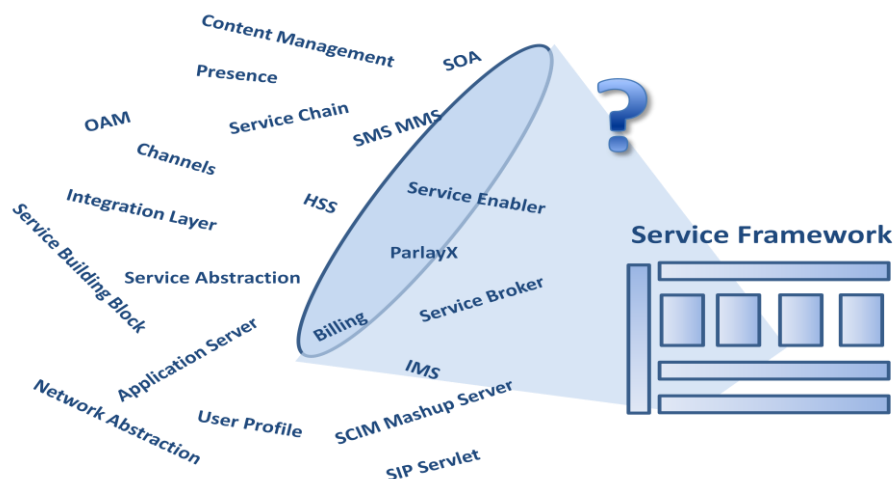
While there are many large vendors that provide Telco type SDPs, some of which have tens if not hundreds of millions of end users, from an information engineering perspective Telco SDPs are tied to the mobile telephone architecture that delivers voice calls, SMS and MMS. Some Telco SDPs also include content and presence services. The Telco SDP data model is centred around the two classic mobile call repositories namely the Home Subscriber Server (HSS) and the Home Location Register (HLR). Both of these repositories are managed by the operator and its OSS functions. The data model is limited with respect to being capable of customer centric self care management and dealing with a full range of on demand converged services.

If the Telco SDP is to increase its that capability for customer centricity, we see that its information model will evolve to that of the MSO SDP version.

### 2.5.2 SDFs Are Vague

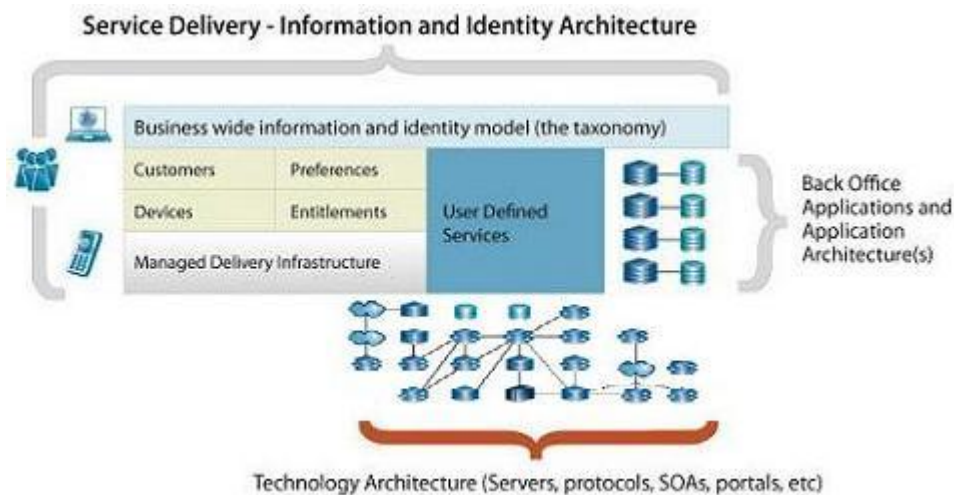
Some of the terms used for Service Delivery Frameworks, Service Delivery Platforms and Service Creation Environments can be very vague. An outcome of such vagueness is the many ways in which SDFs are represented in diagrams, often losing the objective of designing a system that delivers services to customers.

The diagram below identifies the typical block representation of a SDF on the right. However to the left we have many of the well recognized terms being applied. Some of the terms are business functions (e.g. billing), some are technical functions, some are interface standards, some are data stores, some are message/service types, some are abstract IT principles (SOAs) and a few, from an information engineering perspective, are quite unhelpful (e.g. integration layer).



### 2.5.3 MSO SDPs – Information Engineering

In contrast to the Telco SDP, the basic elements of a MSO SDP are identified in the diagram below. Of note is the MSO SDP architecture is an “Information” architecture. The architecture is specified by understanding the information sets and the identity and usage issues associated with such information. Information architectures, once specified are then mapped onto the underlying technologies that can deal with the behavior, scale, capacity, management and security of that information.



Of importance:

- customers and their devices are considered,
- the system design is overseen by an information and identity engineering taxonomy,
- there are five major information design elements namely: customers, devices, preferences, entitlements and user defined services.
- managed delivery infrastructure is included because the MSO delivers managed services.
- and to the right of the diagram the back office applications such as billing, CRM, asset management, Telephone Number management, etc are shown.

The MSO SDP and its functions, characteristics and engineering principles are described in greater detail below.

### 2.5.4 Telco SDPs Evolve to MSO SDPs

As the mobile world converges with the Internet world and systems become more customer centric and self service oriented, we believe that the MSO style of SDP (i.e. wwiteware) will be the SDP of the future.

### 3 Databases and Directories

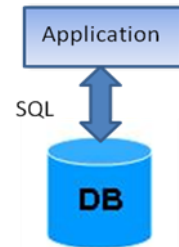
In this section we discuss the differences between databases and directories at the data management level.

#### 3.1 Databases

Databases, particularly relational databases, have been used at the core of all IT systems for decades. The data within them is arranged in one or more “related” tables as columns (data type) and rows (data values). The table, column and row data model is managed and accessed via SQL by the application that sits on top of the database.

The application and its database are designed as a self-contained process with “inseparable and dedicated responsibility” in terms of the operational and business outcome.

The security of the data is enforced through access control schemes that are applied in the context of the application’s transactional design.



#### 3.2 Directories

Directories hold named information objects that are not tied in a proprietary relationship other than name hierarchies referred to as the Directory Information Tree(s) (DITs). Directory objects have a defined class and contain mandatory and optional attributes. The intent of directory (object) store is to hold common information for as many applications and users as possible. Similar to white and yellow page directories, the information in a directory is often considered as the “source of truth” of that identified “resource instance”.

The real benefit of directories comes into play when the directory contains a large number of well managed objects and attributes which are used by many applications for as many reasons as possible.

**It is important to note that** if a directory has a limited number of objects and a limited schema and is tied to one dedicated application, then there is very little difference between it and a similar application using a database.

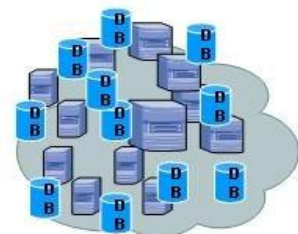


#### 3.3 IT Systems

IT systems comprise many databases and sometimes many directories. Throughout an organisation, distinct sets of databases may be interconnected so that data services can be applied within business groups or across the enterprise.

Normally specific business groups own specific database application processes because the governance has been directed that way. Naturally from an external perspective and for functions that require single view of the customer, the business information system is fragmented, may contain inconsistent data types and possibly has poor online performance.

Where common data has very high demands placed on it, it should reside in one place as that has considerable cost, quality and performance advantages.



### 3.4 Data Models

Critical to both databases and directories are their data models and how they are designed and managed.

While databases and directories have different information properties, it is interesting that database functionality is often evaluated by its data sets - Entity Relationship Diagrams (ERDs), yet directory services are generally associated with the access protocol (LDAP). This perception really started when X.500 – a distributed object oriented directory service with a standard schema set was referred to quite wrongly as “a protocol”.

To normalize this issue we should either refer to all databases as “just SQL” (as per directories - “just LDAP” ) or seek to define and understand the “ERD” of the directory’s identity, information and schema design, particularly when we are dealing with the sharp end of an online business and its service delivery systems.

If the DIT is used for traditional IDM services, it is normal that the DIT naming design follows the organizational structure of a business. The design resembles a white pages service.

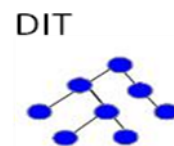
With the larger scale directory services as applied to Multiple Service Operator (MSO) Service Delivery Platforms (SDPs), the DIT designs embrace the front of house needs of the entire user experience as managed and delivered by the organization concerned. In this case the DIT is “quite comprehensive” as it represents the online business – in real time.

Typically such DITs contain subscriber, device, preference and entitlement information.

#### Tables – Columns and Rows of Data



#### Named Hierarchies - Objects containing Attributes



### 3.5 Conclusion

- Databases support process centric transactional data models which are dedicated to the application concerned.
- Directories support identified managed information objects of a common, system wide design that enable multiple applications to operate more efficiently than using dedicated, fragmented data sets.

Where the directory is used to represent the online resources of a business, the directory system scale and capacity represents the scale and capacity of that online business.

## 4 Information Engineering

In this section we discuss the background and key elements of the information engineering required to deliver high performance front of house online systems.

### 4.1 Directories - the identity design of front of house systems

The last few decades have seen a technology and networking focus applied to IT systems, whilst in the background, a distributed information systems evolution has been taking place.

For example (and from experience), the Open Systems Interconnection (OSI) seven layer model, despite being quite wrongly described as “a complex” protocol, was an initiative in the late 1970s by the International Standards Organisation (ISO) and mainframe vendor

application and operating system developers, to get a consistent set of inter-networking service interfaces for distributed applications. OSI was designed as a Service Oriented Architecture (SOA) that applied through profiles, candidate communications and transfer protocols at each layer.

At the time there were about 20 mainframe vendors in the market all with completely different communications architectures (and protocol control languages) which resulted in dedicated software functions and interfaces. Mainframe application customers demanded network **service interface** standards, since porting any application between mainframes was an expensive nightmare. The information engineering exercises of that day were in most part, trying to make networking uniform from an application and data transfer services perspective.

OSI is still used as the model and language to describe the software service layers (and protocol capability) as applied to distributed systems. The ubiquitous "Network and Transport layer Sockets" interface is a prime example of a service-oriented architecture and normalised inter-communication methods.

Since the late 1970s, systems have also evolved from a user-centric perspective. In particular, mobility, the internet, distributed computing and micro technology have played a major part in that. Coupled with the massive increase in compute power and data storage capacity through miniaturization, the major advances in wireless and fiber networking and the World Wide Web, we are now connecting the cyber world with the real one.

**The real issue though, from an information engineering perspective is the real world is not about single users accessing a single portal via a protocol and activating a single threaded transaction onto a dedicated process oriented data base.**

That arrangement is still required, but a bigger picture exists and our information engineering doctrines need to address that picture otherwise the cost and risk with IT investments will increase.

In the early 1980s, networked directory systems came into being and as directories hold **named, identified objects** on scale, directory technologies really started the agenda to identity engineering and how we design front of house information systems for both the user and the business alike.

#### **4.1.1 Identity engineering**

Service delivery within an online system requires that information items representing real world entities and the events from them, are associated in sophisticated and dynamic relationships according to the business and commercial requirements.

Consistency of identity management throughout the system and its external applications and servers is necessary for self care and single view functions. It has the added benefit of reducing the lines of software, numbers of databases, cost and improving service delivery outcomes.

A system design using identity engineering principles begins by defining the types, names and instances of the information items used with a system, understanding the quantities of these identified items (there may be 1 billion items), understanding how their names are used and managed and understanding the rates at which these information items are accessed by the users and the system in the real world.

That is to say, identity engineering is the foundation of information quality control and information performance engineering.

#### **4.1.2 New generation system information taxonomy**

Underpinning customer-centric systems is a system information **taxonomy**. This taxonomy needs to address a wide range of common items such as identity management, products and services, user types, device types, server functions, portal types, locations, content, facilities, call features, roles, error codes, preferences and entitlements, so that information objects,

data tables, integration code, scripting tools and end user interfaces can be coherently specified.

Experience has shown with the larger systems, that each customer-centric information object has at least 100 attributes and each customer can have 50 objects to represent their service needs. Thus each customer could have 5,000 attributes for their online service environments.

### **4.1.3 Information agility**

An implication of technically-focused architectures is that once the system is assembled, any adjustments to functions, data models or the introduction of new services means costly upgrades through an SI project.

The ability to change service-specific data sets can easily be problematic for agile service environments as it means the system adopts a “constant reconstruction” mode of operation.

Experience has shown that changing a single data attribute (and its supporting processes) in an operational system that comprises “data fragments” can take months and cost millions of dollars. There is also the risk that such programs when performed constantly, may not have predictable outcomes.

Information agility is a key requirement of any large commercial online service operation, as is reliability. The information agility issue is complex though as it involves extensive audit trails and many management functions.

However, some aspects of information agility can be solved and features to deal with information agility have been included into the wwiteware engine.

### **4.1.4 Information scale and demands**

It is quite typical for front end systems servicing, say one million customers, to have 50 million objects and up to 5 billion attributes all of which must be accessed in random ways and in real time. This represents a formidable information engineering challenge to any online SDP solution provider.

The information taxonomy has been a difficult road for many organizations, which suffer from a multitude of customised and dedicated data models and dedicated application software. When systems are fragmented and the data models are dedicated, it is very difficult to assess a system for its overall information performance capabilities. To achieve converged services, these dedicated data models need to be realigned to reflect the taxonomy, migrated toward customer-centric management operations – and be applied at the scale and capacity that the online business demands.

This is where wwiteware is able to provide information design machinery to focus and validate such a task.

### **4.1.5 User entitlements**

**User entitlement** defines what a user or device can do on a system from a services perspective.

wwiteware enables rapid entitlement based on product and service sets and provides device, server and portal type catalogues and a system entity model that operates over a top level wwiteware directory information model.

wwiteware also provides a range of system build tools so that all of the above can be loaded into an empty platform and demonstrated very quickly.

#### 4.1.6 Click through management

**Click-through management** of a large scale system is a software engineering architecture in its own right. It relies on coherent information designs being applied to the management functions and the managed entities.

The wwiteware click through management functions are categorized into approximately 80 groups and can be customised and extended. The management functions permit the operator to see, for example, a dynamic information model of the business, view the infrastructure status, see the interface protocol statistics and check the memory buffer allocation profiles of the platform.

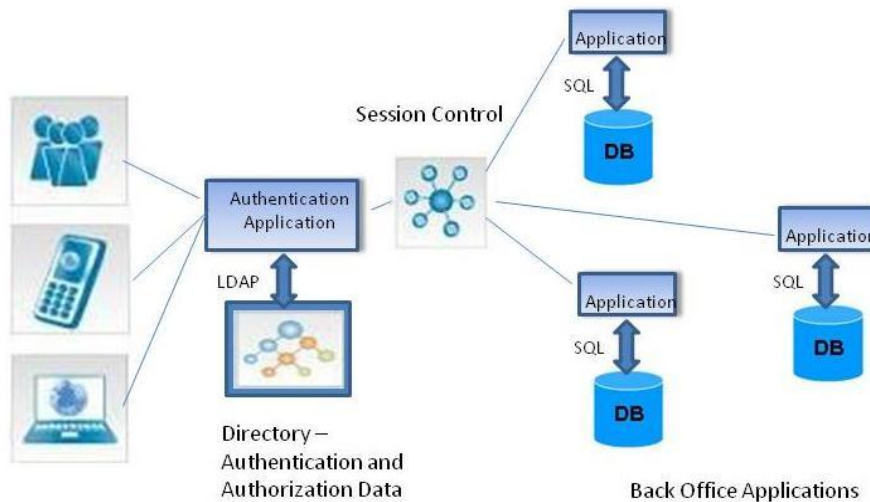
The estimated code base for the wwiteware management system is about 500,000 lines of software. It follows that SDP architectures that don't address identity engineered system management functions present an unknown cost and risk.

## 5 Transactions and Managed Information

### 5.1 The Classic IDM-Directory Architecture

The diagram below identifies the classic "Identity Management" - IDM directory architecture. In this configuration the LDAP server contains the user's authentication and authorization data. The user logs on and their credentials are checked by the authentication application and if valid, the user's access to the back end database applications is enabled.

The diagram unfortunately does not show the DIT design re-naming, objects and attributes or the way it is provisioned and managed, even the non-functional requirements of failover, scale, capacity or throughput are not defined. For such an architecture to be complete and cost justifiable, the information and identity engineering of the directory system and the operational capacity requirements need to be specified.

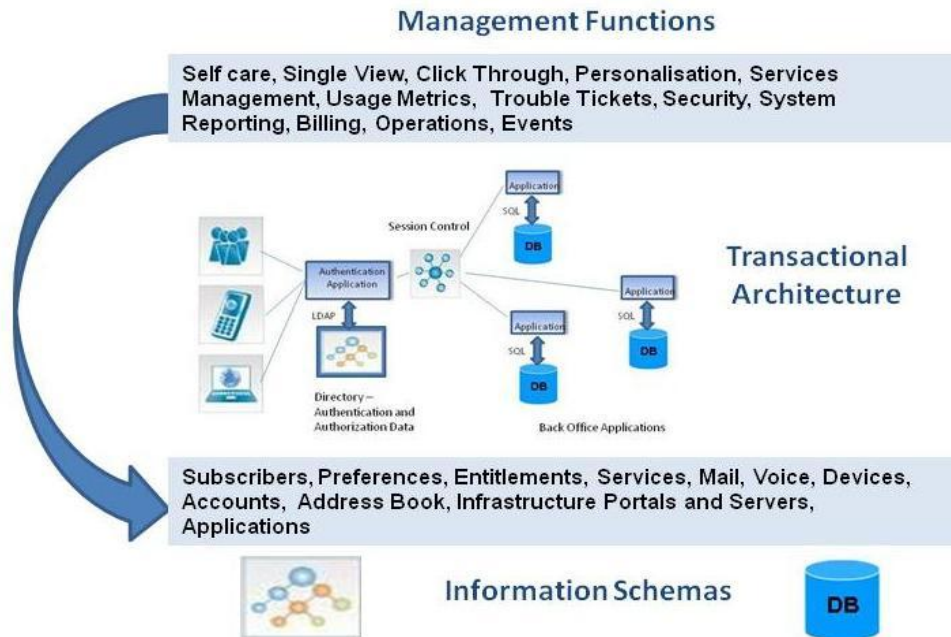


Typically, the very basic procedures around IDM include user enrolment, password management, name checking and life cycle management and these need to be defined upfront as they relate to operational policies.

For example, in many systems if a new user is registered or changes their registered name, the system checks for duplicate names, checks the name to ensure bad language is not used and makes sure that name synchronisation is applied with any other systems.

## 5.2 Adding Information and Management

The diagram below expands on the basic IDM architecture by adding the common system management functions (above) and the related information sets (schema) (below) as typically required by a MSO or large government or commercial organisation.



The diagram identifies, for example, that as user preference and entitlement information has now been incorporated into the system design, it advances the IDM functionality to include Identity Access Management (IAM) processes too.

If we then include “single view” management functions and real time presence services to the system, that implies we need to:

- a) ensure that the entitlement system allows for (a) particular single view(s) of the system and;
- b) for presence, the system must be capable of handling real time user events.

In terms of scale and capacity requirements, we might say at this stage we have a baseline of 1 million users and each user requires 5 information objects in the directory to represent them in the system, therefore the directory system must have the capacity of 5 million objects.

In terms of operational performance, we then might assume that 10,000 users could possibly log in at once, meaning that it is possible that 10,000 access demands are made on our 5 million objects, possibly within a second and if presence services are enabled too, we could be seeing 50,000 presence events at the same time. If the installed system does not meet this basic “information performance” requirement, it may mean that some online users complain or in the worst case, the system fails and then all the users complain.

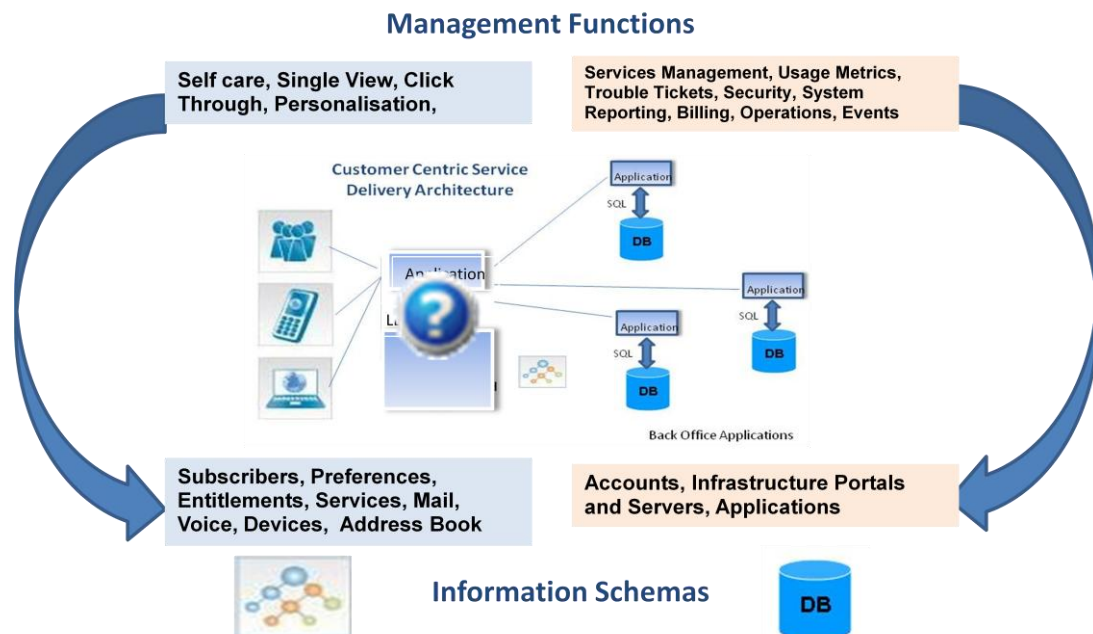
By adding management functions and the information elements to the basic IDM transactional architecture and assessing the real time capacity requirements, we are now addressing the identity engineering and management issues and the operational specification of the entire online system.

## 5.3 The Front and Back Office

An essential step in the overall systems specification process is to delineate between the front and back office data sets and processes. The front and back office model is akin to a

business with a number of retail outlets acting as the front office and the corporate head office performing the back office governance and financial functions.

The diagram below partitions the management functions and information sets of the previous diagram into front and back office contexts.



The diagram also puts a question mark on the IDM-IAM function because for the customer centric functionality described, it will need a significant upgrade.

## 5.4 The SDP – “The IDM-IAM Evolution”

To address customer-centricity, the best practice is to start the system design with the managed services as required and seen by the customer: that is – design the deliverables up front.

The significant advantage of this approach means that the design is focussed on the business, operational and usability of the system in its early stages. In terms of information engineering and service platforms, two dimensions are worthy of consideration.

- SDP functionality within a systems architecture and;
- SDP functionality and its underlying managed infrastructures.

### 5.4.1 SDP Functionality within an Architecture

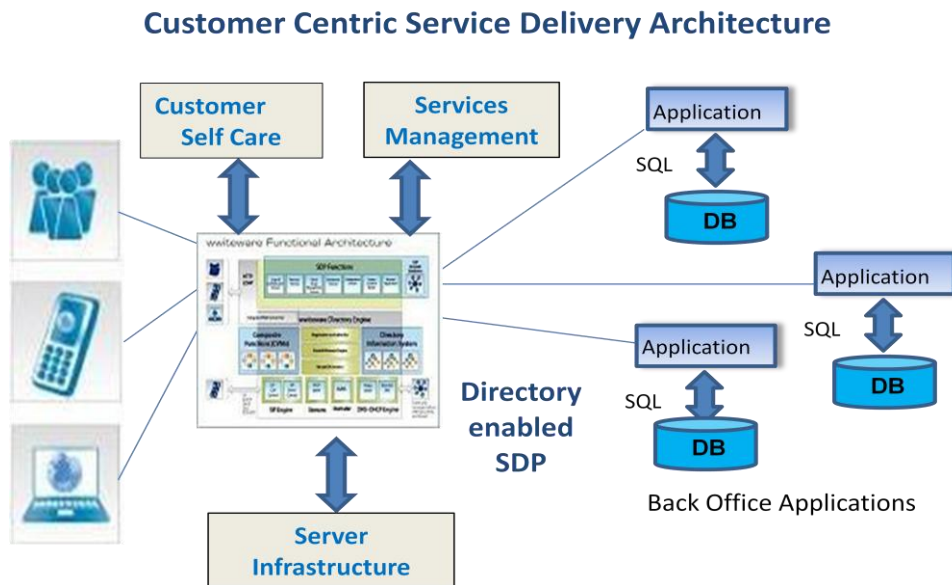
The diagram below identifies how the SDP functionality replaces the traditional (IDM-IAM) authentication function and its “LDAP directory server”.

The diagram identifies that we have also added two critical management functions to the architecture, namely **Customer Self Care** and **Services Management**. Adding these two functions literally directs the system’s information design to provide for single view services and end user entitlements based on user roles and the operator’s services portfolio.

For example:

- The Customer Self Care function may be providing services to Customer Account Executives (CAEs) and the customers themselves (as primary and secondary users of the account).

- The Services Management provides for the product managers of the operator to add or modify their own or third party products and services, which are then seen and selected by the CAEs and customers alike.



### 5.4.2 SDP Functionality and its Underlying Infrastructure

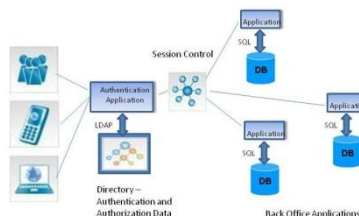
The diagram above also identifies a SDP interface to the underlying server and networking infrastructure. This interface allows the SDP to:

- Assign and provision services onto the respective underlying server functions.
- Monitor the infrastructure status for operability.
- Provide real time single view of the customer’s complete service environment.

Within the SDPs directory engine, objects are placed under, for example, an “Infrastructure” DIT in order for the SDP to have knowledge of the infrastructure it can interface to, monitor and possibly control.

### 5.4.3 Operational Requirements

The IDM transactional architecture shown in the initial diagram implies there is one LDAP server and one authorization function.



Most “architectures” when implemented, have fail over services so the architecture is lacking in this regard. Questions should be asked: What happens if the architecture is applied in every state or in every business facility within a country? What happens if we have 100 portals, 20 mail servers, 15 content servers, 3 billing systems and several customer care help desks in the system? Such questions simply show that “transactional architecture” diagrams have major limitations.

#### 5.4.4 Service Assignment – Infrastructure Identity

With real managed systems we address the “identity” issues of the infrastructure design in terms of what and where the parts are, what their “names” are and how we talk to them to control service delivery.

For example, user management. If a new user is added to the Melbourne office, we need to ensure their online services are not configured into the servers located in Singapore.

Such service delivery and management functions are designed by understanding, at the outset, the identity management issues of the business facility names and places and the system infrastructure components. This design process creates a system-wide identity and information “taxonomy” that will be used by the business for all components in the system.

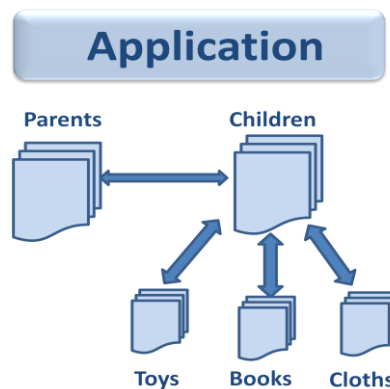
## 6 Data Models, Objects and Their Management

### 6.1 Data models require management

Database models contain one or more tables, where the tables contain rows of attribute values. The column of the table defines the attribute type and the row entry for that column, the data value.

With this approach to system design we know that the application must understand all process issues related to the tables, their relationships, the data types and the data values and manage them accordingly.

A basic database data model is shown below.



The major considerations of the relational data model and its application are:

- The data model design implies THE application will understand every aspect of it and will manage it.
- All the relational and identification aspects of the data are dedicated to the application.
- The application interprets the stored data types in order to deliver a business outcome.
- The application’s design focus may be limited to the transactional process and not necessarily the online end user environment in which it is used.
- Where the data model attributes are associated with the experience of the online user and multiple end user applications are applied across the business, the overall online experience processing issues can become complex.

When developing customer-centric systems, if abstract data models are used, they should include the system and data management requirements and those related to the online user experience and any revenue earning functions. The difficulty is, where there are tens, if not hundreds of databases and data models, which ones deal with the attributes associated with the managed customer centric environment?

## 6.2 Directory Object Management

A Directory object contains attributes, some of which are managed by the directory itself. Every object must have a class attribute, a naming attribute, some operational attributes and possibly access control attributes.

Operational attributes are managed by the directory and indicate the creation or modification date-time of a directory object. The object schema definition also specifies the MUST and MAY contain attributes as used and managed by the directory application. There are only pragmatic limits on how many attributes can be held within an object and how many objects can be held in a directory.

The directory object is really being managed at two levels: the directory manages its object data to ensure the object structure, naming and integrity of the directory is maintained; the directory application manages its data effectively treating the object as a container.

The diagram below depicts a typical directory object with its attribute types (right) and their management responsibility (left). Note the directory service application is managing the non-structural directory object information.

Management Function	Attribute Type
Directory Structure Mgt	Object Class
Application	GUID
Application	Referential
Directory Security Mgt	Access Control
Directory Operational Mgt	Date Created / Modified / Size
Application and Directory Mgt	Distinguished Name
Application	Common Name
Application	User Id
Application	Online Status
Application	Events – e.g. Message Waiting
Application	Image
Application	Entitlements
Application	Password
Application	Attribute Tables

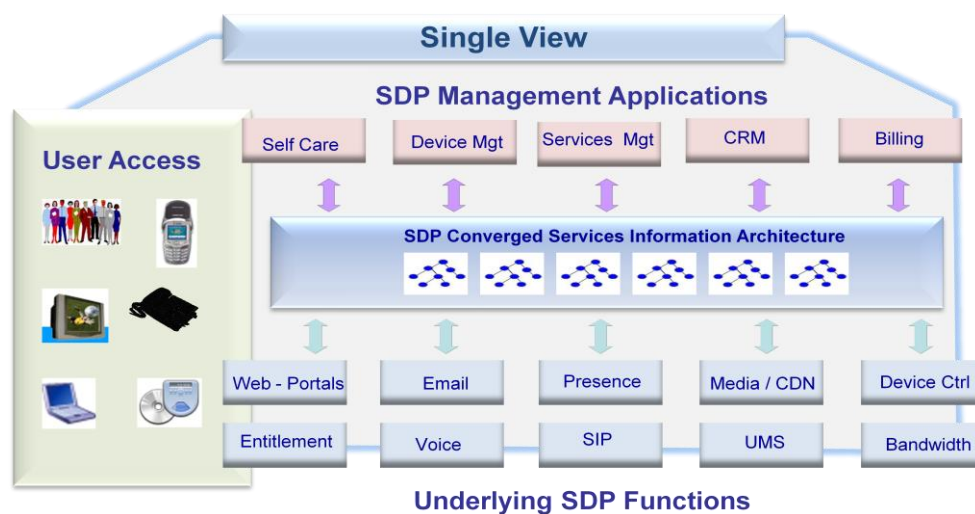
While the directory is being used by or applied to a single application and has a limited schema, this information management model works. The difficulty arises, when the directory contains a considerable number of DITs and object types and requires multiple management contexts. This issue is not directory specific either, the same applies to databases. One application with one database with one data model is fine. Ten applications, with ten

databases and ten data models all to be managed coherently – is a complex if not impossible problem.

The question is – If it is critical to manage the real world issues of online services to identified users and devices with single view functions, etc, the management at the attribute, object, function and system wide levels has to be uniform in that context. So what can be done re our data and object models and their critical need for operational, real time, systems environment management – to make applications simpler?

### 6.3 wwiteware Directory Object Management

The diagram below was provided in the introduction and identifies how the wwiteware SDP management functions such as “single view” applies to a system.



In the customer-centric world, the system, if it is to achieve low TCO, should have a system wide information and identity regime, a common taxonomy, common event handling mechanisms, common message passing, common entitlement regimes and single view services.

In the wwiteware approach, we see that back office and business level applications may want to deal with the dynamics of the real time world, but more often than not, they simply need to only understand a part of it.

Taking the directory object schema above and partitioning the front office real time needs from the application back office needs (in terms of data management) and then allowing the directory to do the common “online world” management functions, we could see the object schema and its management properties as follows.

Management Function	Attribute Type
SDP Directory Structure Mgt	Object Class
SDP Directory Engine or Application	GUID
SDP Directory	Referential
SDP Directory Security Mgt	Access Control
SDP Directory Operational Mgt	Date Created
Application	Distinguished Name
Application	Common Name
Application	User Id
SDP Directory Engine	Online Status
SDP Directory Engine or Application	Events – e.g. Message Waiting
Application	Image
SDP Directory Engine or Application	Entitlements
SDP Directory Engine or Application	Password
Application	Attribute Tables

To this end, the wwiteware engine includes composite functions which understand and apply themselves to “system wide” online environment attributes of the directory objects. Thus relieving any of the applications that use the SDP to have, in part or in whole, the responsibility of dealing with the complex operational, real time world of large scale event driven online systems.

## 6.4 The self-care real time online world

In the self care world, management matters. Data models and transactional architectures and even frameworks may not address this complex but critical area of a (business) system design.

The design of the wwiteware governance engine – the SDP determines and applies the information sets that are generally required for the front of house and applies a range of management functions suited to the self care by customers, the product and service managers and the operational support staff.

The real question is, if this functionality is required and it can be built, will the system be 20 servers and 40 million lines of code and tens of millions of dollars , or will it be just 10% of that.

## 7 The wwiteware revolution

### 7.1 The Front Office

The front office system design needs to be customer-centric. In terms of information engineering, several service dimensions should be considered.

### **7.1.1 Group Identity Management**

In the real world, identified people using identified devices come together to use identified services. It follows that the information that represents these entities, needs to be dynamically related according to service profiles, and managed.

### **7.1.2 Information Environment**

In the online world of the internet, information has a number of forms and processing needs. For instance, an online user or device has a status, we use images, web pages, tables and forms, events are placed on users and devices, there are security attributes applied and of course we use a range of attributes for identity.

However, to build an operational system, we attempt to transcribe this complex set of real world information processing into “transactional data models” - and this is becoming a very difficult and costly process.

### **7.1.3 Presence and Events**

Instant messaging and other forms of status management provide the system, its operators and users with a real time indication of what or who is on line. Presence is a key property of our emerging real time systems such as ehealth out-care, smart cities, smart energy and smart transport.

### **7.1.4 Directory Service Utility**

Directory services can serve a single application with a dedicated data model and the system then relies on other data stores to complete a single activity such as user authorization.

There are examples where fragmented systems use up to 20 directory accesses just to validate a user and up to 80 accesses just to manage passwords. These are extreme cases, but such system designs are destined to fail if they are needed to support large scale ebusiness operations with many millions of customers.

The directory service must provide system level services, not just act as a data repository that holds a simple, dedicated schema and requires full scale searches just to get one attribute.

### **7.1.5 Directory Information Scope**

For real time use, the directory needs to hold all the information that is used to deliver online services in real time. In the SDP case, the directory holds subscriber, device, preferences, entitlements, services and infrastructure information schema.

### **7.1.6 Entitlements and Security**

Security at the front end of a business is centred around “Service Based Access Control” which transcribes to the system’s management of entitlements and roles. Rule Based Access Control is normally associated with process governance or data access protection at the application or technical levels.

All three (Entitlements, Roles and Rules) need to be applied in a coherent manner in customer-centric systems.

### **7.1.7 Devices and Infrastructure**

Whilst traditional IDM has a strong focus on user identity, it should be noted that content, devices, portals and servers as used in the service delivery infrastructure have identity

management requirements too. Service delivery management needs to consider operational location specific and device control information.

### 7.1.8 Information Performance

Whilst traditional architectures portray interconnected functions and “data” transactions, information and identity engineering considers information performance issues at the operational levels at the outset.

Mathematics is often applied to many engineering issues and we do the same with information engineering in order to meet the scale and capacity requirements as an operational outcome.

### 7.1.9 User and Systems Management

- All IT systems need to be managed, and with customer-centric systems it is even more important that management functions are designed up front, since system deployability and usability are implicated (similar to the process of designing a business).

## 7.2 Group Identity Management

The front office design must be customer-centric. In terms of information engineering, two dimensions related to group identity management are worthy of consideration.

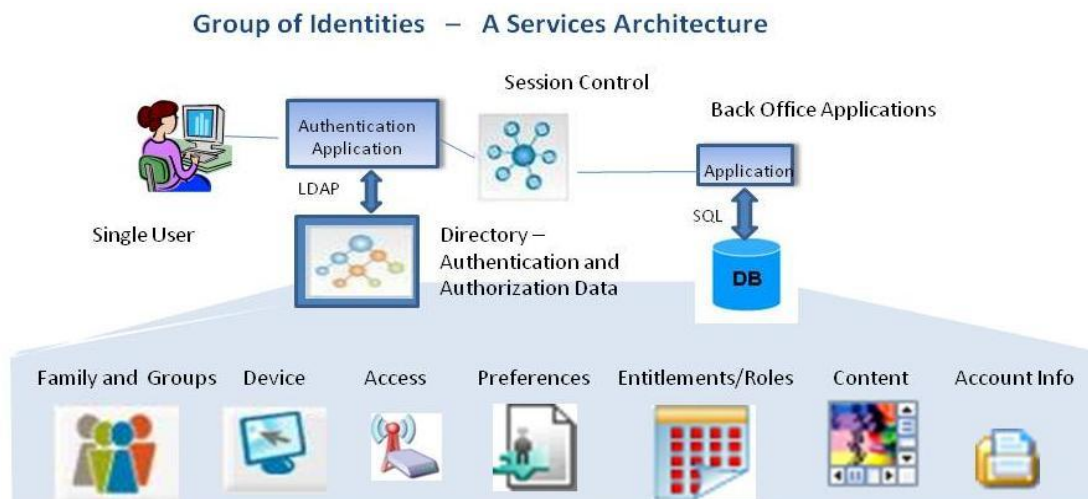
Groups – in real life we either have groups of people and/or groups of entities that we deal with.

- In the first instance a group can be a family where a parent is the account holder and their spouse and siblings are secondary users of that account.
- In the second instance a group can be a single user with devices, content, preferences, access methods, etc.

Groups in either case can be static or dynamic. Groups of identities can exist just for seconds and then be disbanded or modified.

In terms of specifying information architectures, it is good to follow up on how dynamic groups are specified and how they are managed in the operational world, through self care for example.

The diagram below identifies the group identity possibilities and some candidate (grouped) information sets that are used in the online service delivery process.



### 7.3 The Information Environment

The information environment has evolved considerably over the last 10 years. This evolution primarily relates to web services, customer-centricity, user experience requirements and self care.

The traditional approach to IT systems design is to start from the servers, interfaces and transactional data sets and accommodate the end user and device through portals and web services.

The diagram below highlights the system level information management issues that are implied in the transactional architecture's specification.



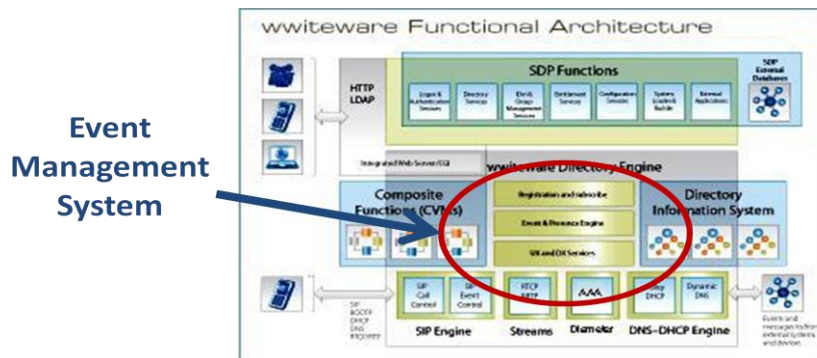
The wwteware directory engine incorporates information treatment functions to deal with the modern day issues of “ end user” information management.

### 7.4 Presence and Events

Instant messaging and other forms of user online status management provide the system, its operators and users with a real time indication of what or who is on line.

The wwteware engine contains the functions to handle presence, online status and an event management system. Adding an event registration and notification service into the directory itself and making this engine the core of a system wide SDP initiative provides significant advantage to the business and its applications over the traditional “data repository” style of directories.

Please note that most architectures do not specify system events (types or priority) nor the event management process functionality needed to deal with them.



## 7.5 Directory Service Utility

Directory service utility is measured by what the LDAP enabled applications need to do to work with the directory. Traditionally LDAP directories are used as data stores possibly just for one application. The directory utility in this case is an object store retrieval system for the dedicated application concerned.

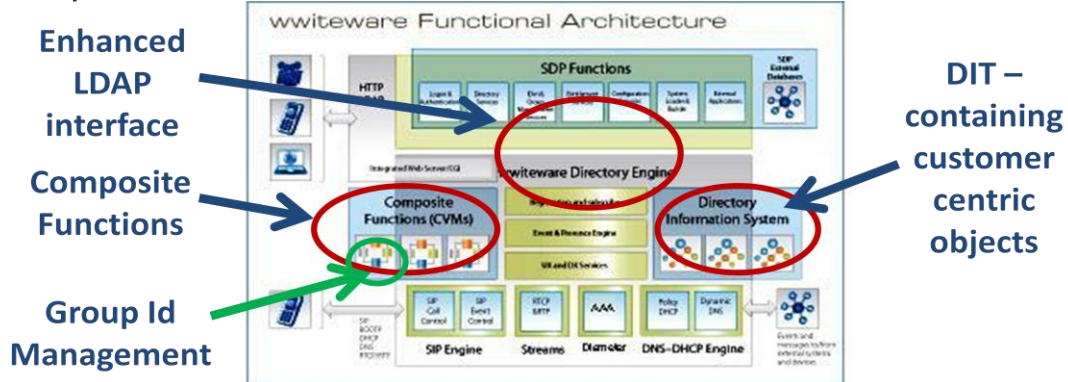
In order to increase the functionality of the directory, more LDAP enabled mechanisms have been developed through the IETF standards process by creating LDAP RFCs. (The list of LDAP RFCs is provided in the following sections). Having a wide range of “protocol” mechanisms means that the application has to support them and use them to its best advantage. However, as the RFC mechanisms are adding more complexity to the directory and its interface, this makes the application more complex – and dedicated.

**witeware takes a different approach that embeds common customer-centric, online services management functions within the directory service.**

For example the “Authorize and Single View “ search, authorizes a user, and if successful, returns the related objects associated with that user, whereas in the basic LDAP world that operation may have required 5-10 distinct Search operations.

**The witeware directory engine has four significant capabilities that increase its “utility” to directory enabled applications:**

- witeware has an enhanced LDAP interface so that its composite functions can be used by the LDAP application.
- witeware contains embedded composite functions to assist applications and therefore simplify application development.
- witeware contains a customer-centric information model (DIT).
- witeware has inbuilt mechanisms that understand dynamically assigned groups of objects.



From section 3.2

*“The real benefit of directories only applies when the directory contains a large number of well managed objects and attributes which are used by many applications for as many reasons as possible.”*

## 7.6 Information Scope

witeware applies information and identity engineering at its core and incorporates a business system information design using multiple, named information trees and an information processing hierarchy for:

- customer features
- the operator’s management needs

- the services and their delivery requirements

The directory engine can manage the relationships between information entities and thus provides a cohesive processing capability for customer-centric, single view, flexible services.

Identity and unique information treatment features in the wwiteware engine, ensure capabilities of high speed, flexibility and high volume processing.

The wwiteware information and identity management hierarchy is represented below.



The diagram shows (left), the major functional entities as represented within and managed by the wwiteware platform. The diagram also shows (centre), the top level naming structures of the directory service. It could be concluded that the directory is just a set of named directory trees of stored directory objects, however, this assumption about wwiteware is quite incorrect.

Each directory tree has a storage and caching philosophy according to its recognised role within an online system. We could call such an innovation **role based information management**.

For example, subscriber and device directory trees are highly cached, entitlements and groups are cached, IMS is a virtual directory of the IMS subsystems, media is a virtual directory of externally stored content and the address book has presence management.

The difference in management philosophies for particular object tree types is the reason for including a complete range of online entity information in the wwiteware platform. These information trees and their related management functions allows a full demonstration of the SDP capabilities out of the box so it can be used as the basis of a front of house system simulator.

## 7.7 Entitlements and Security

Security at the front end of a business is centred around “Service Based Access Control” (SBAC) which transcribes onto the system’s management of entitlements and roles.

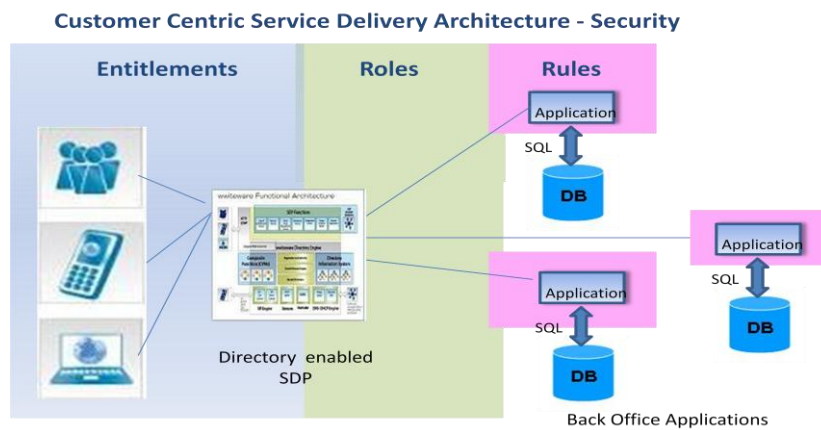
In the larger systems, security is a complex area and is centred around identity (of a device, a user, an algorithm, a token, a policy a role, rule a service or entitlement).

- *Rule Based Access Control* is normally associated with process governance or data access protection at the application or technical levels.

- *Role Based Access Control* is associated to users and devices that can work within or straddle organisation boundaries. Roles can also be applied to customers who may be, for example, a primary account holder or secondaries (spouse or children).
- Roles can be an entitlement, but the point of delineation in customer-centric systems is to use customer roles (account types) and their services as entitlements. Staff on the other have their duties related roles. However, the line between roles and entitlements can be unclear.
- Entitlements are applied in the wwiteware directory engine and in its management functions. Entitlements are considered in three main groups, namely: customer, product manager and operational support. Other groupings are possible within the wwiteware platform in order to support different types of business requirements. Entitlements can be configured and then presented for selection from which entitlement object is written to the wwiteware directory engine. wwiteware contains a number of entitlement management functions and caches for rapid processing.

The principle issue with defining entitlements in a customer-centric environment is that the language of all the entitlements represents the service language as used by the CIO and the customers. (Not language associated with technology terms).

All three access control methods (Entitlements, Roles and Rules) need to be applied in large scale service delivery and customer-centric systems.



## 7.8 Devices and Infrastructure

Device and infrastructure management is also critical to customer-centric service delivery systems, as in some cases, devices such as modems and set top boxes (as well as interconnected servers) also authorize themselves on a system the same way human users do.

The wwiteware engine holds configurable catalogues of device, portal and server types and these are used to create information objects in the directory that represent these entities; ie information that represents the access and network services infrastructure.

These catalogues are identified in the diagram below.



The wwiteware engine also contains device management caches and some unique features around its information treatment algorithms such as event I/O processing.

In traditional systems, provisioning applications write information representing the server into the directory and then subsequently write it to the server itself - two very distinct application actions for every resource in the system.

The wwiteware engine has the ability to classify attributes for event management and when these are written or read, a down-stream action will occur to or from the real server that the object represents. This feature simplifies management and provisioning application development and operations.

## 7.9 Information Performance

Information performance specifications are critical with large scale real time customer-centric systems. The user's online service expectation is quite selfish at times. Regardless of anyone else on the system, the system must be instantaneous! If it does not meet this requirement, four actions are possible.

- Patience – and the user comes back later
- Complaint – at a cost to the operator
- Complaint and leave – at a cost to the operator
- No complaint - but simply go to another operator

These are not good outcomes particularly if all that is noticed is falling online revenues.

Basic examples of information performance calculations are:

- 10 million users with 10 million devices that require 5 information objects each, mean the system must have a capacity for 100 million objects and be able to manage the names/identities of such.
- As above but 100,000 users may log in at the same time, meaning that 50-100 million objects may be searched through 100,000 times a second to perform such tasks.
- As above but each object can have 20 – 40 attributes each. The system must be able to store and access 2-4 billion attributes.

## 7.10 User and Systems Management

wwiteware addresses user management and system usability as a foundation design element.

- Experience shows that a transactional application with its data model design might only represent 50% of the total code base to that of a well managed system that has click through, single view features.
- While it is necessary to authenticate users and devices on a system, it is equally critical to have consistency on what authenticated and authorised users do beyond that of a transaction on a database. Well managed entitlements are key to defining what users and devices do once authenticated.
- Management functions also include load and build test tools, archive and back up tools, specific functions for devices, the technology itself and of course logs and traces.
- wwiteware applies click through, single view management functions as a critical part of its information design architecture and they are contained in multiple directories within the SDP. The sub-directories are organized based on the respective platform function and can contain frame, html, dedicated scripts, images, configuration information and tables.

The table below identifies some of the management functions of the wwiteware platform. While not all are needed in all systems , it is recommended that all architectures identify the management functions required for the architecture to operate.

Function	Function	Function
Applications	DX Management	Product Logon
Architecture Help	Entitlement Management	Read - Delete
Authorization Svcs	Entity Fn Help	Roll Over
Build Devices	Entity Help	Scripts JS
Build Infrastructure	Entity Management	SDP Logon
Build Products	Entitlement Logon	Search Directory
Build Subscribers	Entry Management	Session Control
Build System	Event Management	Session Management
CADS Management	Event Log Management	SNMP Management
CADS SDP	File Management	Styles CSS
CADS Configuration	Forms Management	Subscriber Management
CADS Clone Control	Group ID Management	Superior Info
Common Data	IMS Management	Services Management
Content Management	Infrastructure Config	Services AC Management
Console Management	Infrastructure Management	System Management
Defaults	Inode Management	Table Management
Delivery Control	LDIF Tools	Transaction Control
Device Management	SDP Libraries	Transaction Images
DHCP Management	Log Management	Transmit Message
Diameter	User Logon	User Management
Directory Control	Manage Info	Utilities
Display DIT Utilities	Manage DIT	UX Management
DIT DriveMap Management	Modify Entry	User Entl
DNS Management	Move Entry	Web Pulse Management
Data Preparation	Inode Management	Web Pulse Configuration
Drive Control	System Poll	

## 8 Conclusion

The wwiteware engine represents nearly 40 years of experience in designing, building and deploying online systems from an information and identity engineering and services delivery perspective. The demands of modern commercial online customer centric business systems require that both architects and engineers address scale, capacity, usability and profitability as a key design requirement.

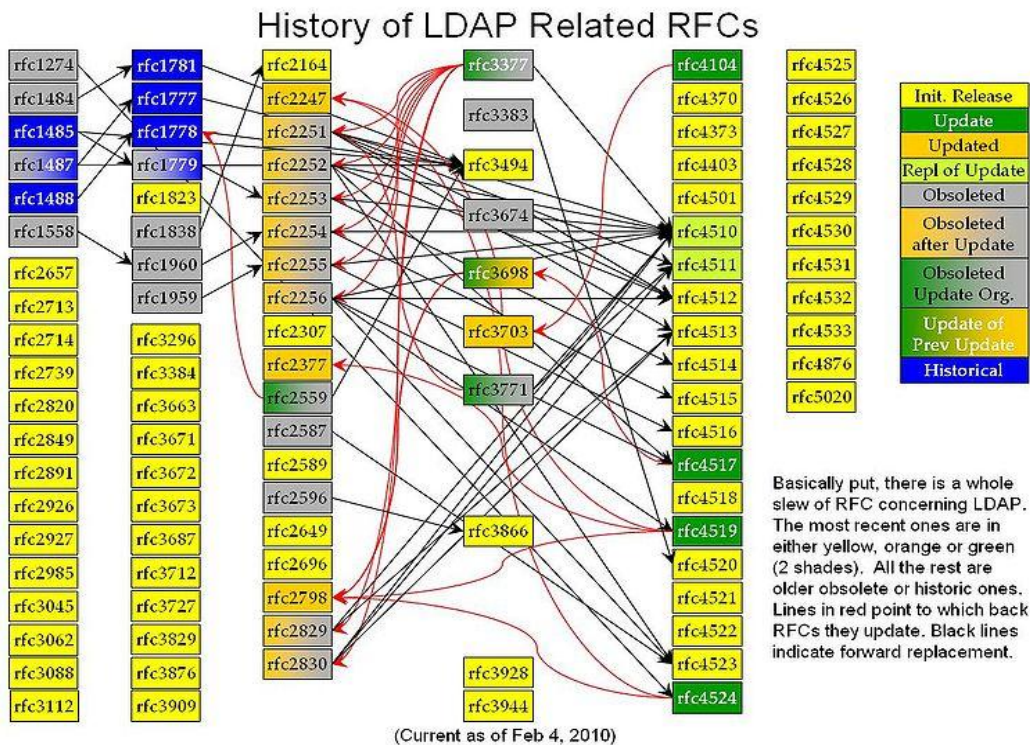
Additional wwiteware information provided at [www.wwite.com](http://www.wwite.com)

## Appendix 1 - The Application of LDAP

### Current LDAP Status:

There are many LDAP RFCs that provide interface control mechanisms that enable the LDAP application to access and manage the LDAP data store. We note that Wikipedia <http://en.wikipedia.org/wiki/LDAP> provides a LDAP RFC chart (below). In terms of currency or obsolescence we note that the RFC 5020 (the latest LDAP RFC) has a publish date of 2007, with most other latest LDAP RFCs having a publish date 2006.

Chart : Courtesy Wikipedia



## **LDAP Issue: Micro-Mechanisms**

LDAP represents an evolution from the X.500 DAP (X.511) circa 1985. However, over the last decade LDAP has become “mechanism rich” but at the same time at the operational level LDAP system implementations could be considered as “application poor”. Application poor means that most LDAP servers are used as a data repository to hold a simple schema for “Identity Management - log on” or organizational white page services applications and the like, Typically LDAP servers are configured with a dedicated schema of 20-40 attributes per object- user.

Having this “multiple LDAP RFC - mechanism rich” approach to protocol design means that the system application and its data model are being designed at the seriously detailed interface and data attribute level – based on “a data storage and retrieval capability” which probably excludes end user service functionality or a customer centric outcome.

It is quite possible that considerable time and cost will be incurred finding out how the application might use a particular LDAP feature and a few data items to its advantage. This design approach can get right down to the single attribute level and how that attribute is managed. For systems with thousands of attributes and millions of objects representing millions of users and devices – this design approach could take many years and many millions of dollars.

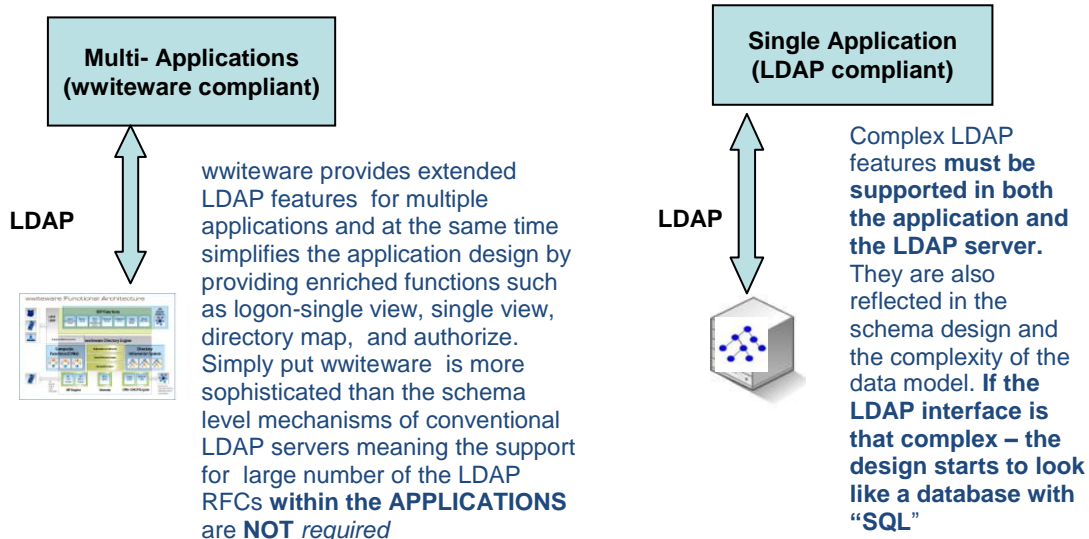
In applying LDAP – the following questions should be asked:

- Are all of the features provided in all the LDAP RFCs being used for a service delivery advantage?
- Are these features only being used by one particular application?
- Is the application being micro engineered to suit the micro engineered features of the LDAP server?
- How complex should the LDAP interface be?

Perhaps the answer to the last question is: It should be as complex as the public domain LDAP scripting tools allow E.g as supported by PERL Graham Barr <gbarr@pobox.com>. package Net::LDAP

If the LDAP interface is too complex – it means specialised and dedicated schema design, application development and support skills are being applied at a cost.

*wwite suggests that in development projects the LDAP RFC list is initially put to one side until the system's information (schema) engineering designs have been completed.*



*“The wwitware agenda is to simplify the complete system design , its implementation and its cost by providing managed service based technology which can be demonstrated immediately, easily applied to application development and evolved into operations.*

*The LDAP server and RFC approach – is literally starting a large scale, customer centric, business system design from a data repository protocol, possibly even down to the single data attribute level... That is: Starting the system design at “ground zero”.*

## Contact us

Susan Oliver tel: 61 408 070 071 email: susan.oliver@wwite.com  
 Alan Lloyd tel: 61 418 536 749 email: alan.lloyd@wwite.com

wwite P/L © 2008



## Disclaimer

The information provided in this document is for use of a general nature only and is not intended to be relied upon as, nor to be a substitute for, specific professional advice. No responsibility for loss occasioned to any persons acting on or refraining from action as a result of any material in this publication can be accepted.