



REORGANIZATION OF TRAVEL DISTRIBUTION SERVICES GROUP

2005 COMPUTERWORLD HONORS CASE STUDY

BUSINESS & RELATED SERVICES

THE DEVELOPMENT OF THE TRAVEL INDUSTRY'S FIRST SERVICE ORIENTED ARCHITECTURE FRAMEWORK AT CENDANT TRAVEL DISTRIBUTION SERVICES, WITH ITS CREATIVE CENTRALIZATION OF KEY RESOURCES AND STANDARDIZATION OF HARDWARE BOTH IMPROVED SERVICE AND YIELDED MORE THAN \$100 MILLION IN COST-AVOIDANCE BENEFITS. [20055185]

A Search for New Services



SUMMARY

The responses below describe the successful development of a culture of reuse within Cendant TDS beginning with the centralization of key resources, followed by hardware standardization which delivered cost avoidance benefits of over \$100 million and culminated in the development of the industry's first Service Oriented Architecture framework at TDS.

Robert Carrigan,
Chairman of the Chairmen's Committee

Ron Milton,
Vice-Chairman of the Chairmen's Committee

Dan Morrow,
Chief Historian

APPLICATION

Cendant Travel Distribution Services (TDS), a subsidiary of Cendant Corporation (NYSE:CD), is one of the world's largest and most geographically diverse collections of travel brands and distribution businesses. The division includes: Galileo, a leading global distribution services (GDS) company, serving more than 44,000 travel agencies and over 60,000 hotels; hotel distribution and services businesses (TRUST, THOR, WizCom and Neat Group); leading online travel agencies (Orbitz, CheapTickets®, Lodging.com, HotelClub.com and RatesToGo.com); Shepherd Systems, an airline market intelligence company; Travelwire, an international travel technology and software company; Travel 2/Travel 4, a leading international provider of long-haul air travel and travel product consolidator; online global corporate travel management solutions, through Travelport and Orbitz for Business. TDS connects the most buyers and sellers in the global travel marketplace.

The TDS Data Center handles, on average, more than 350 million messages per day, 4051 messages per second and more than 912 million bookings/cancellations a year and more than 206 billion fare quotes per year.

Acquired by Cendant in 2001, Galileo International is known to have the most efficient development staff in the business. Although the vast majority of these technicians have grown up in the transaction-processing facility (TPF) mainframe world, (TPF being the primary processing system of all major travel companies capable of processing upwards of 8,000 messages per second) Galileo has long lead the industry in efforts that opened up their legacy systems to outside customers.

In the 1990s Galileo International became the first GDS to open up access to its systems via program friendly interfaces known as structured data upon which many of today's leading travel products were built and still operate. In the years that followed, Galileo extended this capability by offering interfaces in the (then) emerging standard of eXtensible Mark-up Language (XML). Then again in 2002 Galileo lead all industries by offering the first real-time, transaction-updateable interfaces via SOAP standard Web Services.

With Galileo Web Services, customers could dramatically shorten the amount of development time it took to access the rich features and functions that Galileo had developed over decades in the travel industry. In some cases development time was reduced by 90% by coding to the Web services interface rather than the traditional methods offered by its competitors.

Around the same time, Galileo was embarking on a bold mission to become the world's first GDS to completely rewrite and migrate its airline fare shopping and pricing system from a proprietary mainframe environment to a distributed UNIX system.

In the midst of these exciting advances, the travel industry was facing uncertain times with revenues sharply down following terrorist attacks in the US and Bali and fears of a SARS epidemic in Asia.

The new Cendant organization understood that it was critical for to build a strong architecture team that could drive the right technological solutions while we let the application designers and developers do their magic.

The first thing we did was to take key service resources (architects, DBAs etc.) spread throughout the company in different groups and centralize them into one team. The benefits being that they would not only learn from each other but also so they could be deployed to the projects with the highest priorities rather than the groups that had arbitrarily hired them.

We then strengthened these teams with individuals from outside of the company (Sun, Delta, IBM etc.) that could have responsibility for key technological verticals (hardware, OS, database, storage etc.). Although everyone on the team would have an opportunity voice their opinions – these individuals (we called them X-Men although, like the comic heroes, they also included women) would be the go-to people for the areas they were responsible.

With the extended team in place we assumed responsibility for all architecture decisions across first Galileo and then TDS as a whole. Things like: choice of database platform, which application servers to buy and what OS to run it on and so on. By moving this responsibility from the project teams to a centralized group of specialists we made both teams more nimble. The project teams could focus on the business logic and the application architecture while the infrastructure and system architecture was handled centrally.

The architecture team's overall responsibility was to focus on reducing Total Cost of Ownership (TCO). This formula had four parts. The most important one – the one we would not sacrifice – was stability.

Our analysis proved that we could get extremely high levels of stability by architecting our solutions with lots of high performing, horizontally scalable, redundant servers. In 2002 we partnered with IBM to establish the first of our repeatable hardware configurations we called our cookie-cutter-architecture. This architecture was based on IBM Intel processors and gave us benefits that were so compelling that we abandoned UNIX as the target for our new Fares system and migrated the system to Intel.

A price per performance comparison between the two technologies showed that the Intel machines were 3X faster and 90% cheaper than their RISC counterparts. Furthermore, having Intel as our platform of choice gave us much deeper cost savings versus our proprietary mainframe system at a time when Internet driven air-shop to air-book ratios were rising sharply. Recent financial estimates have shown that by moving to this new environment saved over \$100 million in 3 years – versus what we would have paid on the mainframe.

The cookie-cutter-architecture (which has since been extended to blade servers with Network Attached Storage (NAS) devices for greater flexibility, greater redundancy and simpler maintenance) also gave us other key benefits:

- Greater core technical competencies
 - o Technicians (Operations, Admins etc.) could develop deeper skills on a standard solution set. Repeatable systems could be installed much faster
- Greater Reuse
 - o Performance test, staging and copy environments – even Disaster Recover complexes – could be shared by multiple systems
- Greater bulk purchasing deals etc.
- Faster time to market
- o Not having to reinvent the wheel every time greatly improved project efficiencies

In 2002, Galileo successfully launched its Web services. For customers who were beginning to create a Web presence for themselves, the attraction of Web services was very appealing. Our customers realised that most of the functionality they would need to develop to create an Internet booking engine (IBE) to launch their product would be complex and non-differentiating.

Think about the last time you went to a travel Web site to make a reservation – did you go there because it logs errors well, or because it retrieves your customer file in a particularly cool way, or even because it uses one database vendor over another? Maybe - but most of the time it was probably because it displays information well, offers you some nice features and – most importantly – found you the lowest price for the trip you want to take.

The problem of duplication at all levels of IT is that 70% of the work done by an average company is focused on design, development and maintenance of commodity-type functionality. While they are no doubt important, in fact they are critical to the successful execution of the product, they do not create a compelling reason for a customer to choose one product over another.

What does make that difference is the other 30% - and it is the goal of maximizing software reuse through our Service Oriented Architecture that allows our business units and our agency customers to focus their intellectual skills on the portion of their products that do create differentiation; i.e. presentation, exciting new features and – above all – the content (the right product at the right price) consumers are looking for.

Strongly committed to a new culture of standardization and reuse, along with a belief that huge benefits can be gained by allowing customers and other business units to eliminate redundant development by offering dynamically configurable services, the TDS Architecture team embarked on an effort to leverage the advantages of its Web services and evolve them into the industry's first Service Oriented Architecture (SOA) framework.

This framework allows small logical software programs (services) to be rapidly assembled into larger composite-services customized to perform specific tasks (shop for a flight, book a flight, shop for a car and hotel, show me pictures of selected offerings, book a flight, car and hotel etc.) for many different products, business units and customers.

BENEFITS

- Has your project helped those it was designed to help?

Yes, in every area.

Technologies will change. What is cool today will almost certainly be passé tomorrow. In order to protect our investments it was critical for us to implement and enforce standards and procedures that insulate and abstract our human and technical resources as much as possible from the details of the inevitable changes to the current technology-du-jour.

The full focus of our initiative was to reduce redundancy at all levels across the IT lifecycle, freeing up key technical resources so they could focus on the key differentiators and avoid huge wasted costs.

- In your opinion how has it affected them?

Development teams became much more productive as we moved all vendor platform decisions (which database vendor, what OS etc.) to the architecture team. This allowed the project teams to focus on customer requirements and business logic design and flow. In 2003 alone, this simple move resulted in faster time to market and cost avoidance savings of over \$6 million.

Standardizing on a cookie-cutter-architecture, based on IBM Lintel servers, and enforcing vendor agnostic strategies - such as developing data abstraction layers for easier migration to other solutions at a time when they offer better product value, standardizing programming languages that will port to other technologies if and when Lintel (X86) processes no longer offer the best price for performance – we continued to see huge benefits of standardization and reuse.

The hardware standardization strategy allowed us to get greater reuse of our test systems and share our Disaster Recovery environments. Also, our technicians began to develop deeper core competencies as we began to reduce the number of technical platforms we supported – and our purchasing department was able to structure more attractive deals through bulk purchasing across multiple systems.

Moving our Fares system from a proprietary mainframe system to over a hundred, high speed, highly redundant Lintel systems presented us with cost avoidance savings of \$100 million over 3 years.

The software focus on reusability came from the development of our Service Oriented Architecture framework. Analysis showed us that developing reusable components added about 50% to the development lifecycle (design, testing, etc.) but having only two users of the same function quickly put these efforts into the black.

- What new advantage or opportunities does your project provide to people?

Often the benefits required not only an understanding of what currently is, but also what could be.

For example - one of the biggest differences between a travel product built for business travelers and one built for leisure travelers is the Corporate Travel Policy. This is a set of corporate level rules that tell the booking

logic which travel vendors the company has deals with and wants their employees to use. Based on these rules, the booking engine logic will select travel vendors that comply with these rules – e.g. Fly United Airlines, stay at Ramada Hotels etc.

On the surface this requirement does not exist for leisure travelers – but a closer inspection shows that leisure travelers often have a preference through loyalty program memberships or personal experience and having an option to pre-filter their selections would be very useful to them.

Another big difference between business and leisure travelers is that the former is more likely to be schedule - focused (i.e. fewer connections, shorter travel time) while the latter is predominantly price-focused (i.e. possibly more connections, provided they are compensated with an adequate reduction in price).

Again, on the surface, this difference appears to be significant but in reality it may (and should – if designed correctly) be a simple difference in a parameter sent to the air fare search tool. After all, business travelers and leisure travelers are not different people, they are the same people traveling for different reasons and – in the end – their goals may overlap considerably.

- Has your project fundamentally changed how tasks are performed?

Yes. There has been much more focus on assigning tasks to specialists. Cendant TDS has a substantial IT infrastructure and budget and so centralizing decisions on database vendors for example greatly reduces the likelihood that we will increase our vendor ‘footprint’ without the necessary people agreeing that the decision is strategically aligned and that the necessary due diligence (performance tests etc.) has been done. It also reduces the likelihood that one group will buy new licenses for a product that another group already owns.

Software development goes through a very similar process. Traditionally the decision on whether a piece of software should be reused or extended – versus being written from scratch – is left to the developer. This is exactly the wrong person to be making this decision. Developers like to develop – that’s what they do. Developers will almost always write from scratch.

Access to our SOA framework is made via an XML layer, Business Domain Model we call the ‘TDS Rosetta-Stone’. Just as the original Rosetta-Stone made an old language readable by translating into a ‘modern day’ language, our TDS Rosetta-Stone translates disparate interfaces from multiple content providers (old and new) into standard XML.

Before a programmer can deploy a new service into our TDS SOA framework, they must first request an update to TDS Rosetta Stone supplying the name of their new service along with the input and output parameters it is associated with. This is reviewed by our librarians and if there is an overlap between the parameters (and/or the description) of this new service and one that already exists, further investigation is done and a more informed decision is made as to whether or not a brand new service is justified.

The creation of a world-class test lab and an outstanding performance test team has also helped us create highly efficient and high performing systems, with plenty of data as to their scalability potential and breakpoints so that we predict their behaviors and respond to them before they impact end users.

- Does your work define new challenges for society? If so please describe what you believe they may be.

The Internet is still a very new technology, one that has become so pervasive in our lives that we often forget that as a true commercial environment it is less than ten years old. Much of what we, as consumers, deal with is still very immature. For example, most Web sites offer pages that are really nothing more than an electronic representation of a paper medium like a simple pen and ink form.

Greater reuse of commodity services will allow more attention to be devoted to the evolution of competitive differentiators such as content and presentation.

The faster the industry embraces this type of reuse, the faster we will unburden the inventive thought processes of those individuals that can advance these products, providing richer, more interactive, graphical interfaces – accessing a far wider range of content serving more international languages and offering product marketing opportunities we cannot yet imagine.

IMPORTANCE

- How did information technology contribute to this project?

By leveraging the skills of very talented individuals, with experience spanning decades of mainframe, UNIX and other open systems technologies, the team was able to leverage their knowledge and to utilize the best of the emerging technologies to create great advances across our organization.

The SOA reference implementation - a lightweight product capable of rapidly processing stand alone business components in order to satisfy the needs of the user – was extremely successful attempt to apply the concepts of reuse and standardization to software services. The names of the individuals who were primarily responsible for that reference implementation are as follows:

Glenn Harper Principal Engineer
Bryan Harwood Principal Engineer
Nick Josephs / Sharon Doyle Project Managers
Jennifer Morgan Lead Architect
Steve Schoenstein Lead Architect
Glen Zwart Principal Engineer
Robert Wiseman CTO / Project Sponsor
Mickey Lutz CIO

The primary goal of the initial architecture team was to create a flexible reference implementation that proved a working/functional framework could be designed around the following core services:

- Business Process Service – workflow management
- Caching Service – temporary storage
- Data Access Service – database operations
- Logging Service – transactional, diagnostic, and informational logging
- Messaging Service – Java enterprise messaging functions
- Transcoding Service – transformation between XML and Java objects and between different XML schemas

One example of the flexibility that was inherent in the product's design was that all data access it performed through the DAS layer, abstracting the proprietary details of the underlying data structure. This capability allows us to more easily replace supporting databases without requiring changes to the calling services.

In fact the entire framework was designed so that all of its key components could be switched out as new and better capabilities became available. In fact many of the specific products that were used to make the reference implementation have already been upgraded or simply replaced to increase the robustness, scalability and/or overall performance.

Also inherent in the framework's design was an awareness that services may reside locally or externally and may be built with Java and non-Java (.Net, Assembler etc.) technologies - that external services had to be called via an XML interface but wherever possible, it would be more efficient to use local Java calls if and as appropriate.

Furthermore, in order to provide maximum flexibility and scalability, it was necessary that the calling service should never need to know the location (or interface – Java or XML) of the service it is calling.

All this is accomplished through the use of a service broker via which all service to service calls are made. Each time a call is made to the service broker it does a look up via an LDAP directory to find the location and contract requirements of the target service. Then, using Exolab's Castor product, the service broker can – if necessary– transform data from the format of the calling service into the compatible format of the target (i.e. from Java to XML or vice versa).

This past year, the team itself was enhanced by the addition of two new Principal Engineers, Bob Roth, who has assumed responsibility for continued evolution of product and Beckie Watson who has responsibilities for Enterprise Performance Testing.

With Bob's assistance we have further extend the TDS SOA framework's capabilities by incorporating an Enterprise Service Bus (ESB) solution to provide large scale content aggregation, coupled with orchestration and rules services to provide rapid and intelligent access to disparate content providers – a key differentiator for travel distribution products.

- Why was information technology particularly important to it?

Access to high quality products from the Open Source community allowed the team to construct a working reference model at a relatively low cost. The ability to leverage these types of tools is something that has only become available within the past decade marks an important advancement that greatly benefits designers of IT products worldwide continuing the the acceleration the high quality of products at an unparalleled rate.

The TDS SOA design team used a mixture of Open Source and third party products to develop the working concepts of its reference implementation. Though many of these products have since been replaced (a capability provided by the componentized design of the framework), their availability at the early stages of development were invaluable in proving out the products potential.

- In your opinion, have you developed a technology that may lead to new ways of communicating or processing information?

Yes. The Service Level Manager capability allows the product to dynamically measure service responses and compare them to Service Level Agreements of its customers. For example, if we have an agreement to provide responses to a given customer within 2 seconds, and the Service Level Manager 'sees' that due to heavy loads on the system we are in danger of failing to meet that SLA, it can do one of two things – either a.) increase the computer resources dedicated to the services the customer is using, or, b.) redirect all of the requests for that customer so that they have dedicated resources to themselves.

- Describe any new technologies and/or cite innovative ways to use existing technology to create benefits for society? Or did you define a problem and develop new technology to solve it?

In addition to the ESB (described above), we have tested out and successfully implemented a new line of products known as XML Appliances. Contrary to our cookie-cutter-architecture approach where we have repeatable configurations of hardware that can be shared across multiple types of disparate applications, the XML Appliances only support one type of application – XML transformation – but they perform that task very quickly. In some cases 10 to 30 times faster than performing that task using application code running in traditional application servers.

In 3Q04 we implemented these devices into our Cheaptickets.com environment and saw dramatic improvements in site speed. In 1Q05 we will replace the XML gateway of our TDS SOA framework with an appliance that provides wire-speed authentication and XML transformation giving us the ability to support multiple XML formats with no discernable loss in performance.

Also, see SLM (above)

ORIGINALITY

- What are the exceptional aspects of your project?

The systematic implementation of high quality standard procedures which was successful in creating a culture of teamwork and reuse generated numerous exceptional aspects worth recognition. The TDS SOA framework probably delivered the most notable items, namely:

- o Service Level Manager – capable of dynamically reallocating stressed resources
- o Business Domain Model/Rosetta Stone – the façade to our services
- o Service Broker – key to execution efficiency by ensuring that internal calls can be executed without the unnecessary overhead of XML
- o Data Access Services – initially a data abstraction layer, will soon play a greater role in detecting database outages more quickly than traditional HA solutions

Other notable mentions belong to the cookie-cutter-architecture, the industry leading move to Network attached storage - and the a graph (which I was unable to insert into this response), which very nicely allows us to predict global resource usage. Taking a 24 hour resource usage graph and lining it up with a map of the world - showing time running from right to left (to follow the Sun) and lining up any hour on the time scale with the time zone associated with it (in our case Mountain Time) we can see how volumes for that hour of day, anywhere in the world. For example, 1500 hrs is over MST and so each of the 24 axis points shows the system volumes as they relate to that time in every country in the world – e.g. 1500 hrs in England, 1500 hrs in India etc.

- Is it original? Is it the first, the only, the best or the most effective application of its kind?

So many features delivered under this strategy are either leading edge (move to NAS, XML Appliances, Cookie-Cutter-Architecture) or the first, or among the first, in the industry (SLM, Fares to Intel, the resource map) that the overall strategy has to be considered to be at least one of the most effective and successful applications of its kind

- How did your project evolve? What is its background?

In the 1990s Galileo International became the first GDS to open up access to its systems via program friendly interfaces known as structured data upon which many of today's leading travel products were built and still operate. In the years that followed, Galileo extended this capability by offering interfaces in the (then) emerging standard of XML. Then again in 2002 Galileo lead all industries by offering the first real-time, transaction-updateable interfaces via SOAP standard Web Services.

With Galileo Web Services, customers could dramatically shorten the amount of development time it took to access the rich features and functions that Galileo had developed over decades in the travel industry. In some cases development time was reduced by 90% by coding to the web services interface rather than the traditional methods offered by its competitors.

Strongly committed to the idea that huge benefits can be eliminating redundant development by identifying and sharing commodity services, the TDS Architecture team embarked on an effort to leverage the advantages of its web services and evolve them into the industry's first Service Oriented Architecture (SOA) framework – a software architecture that allows small logical software components to be rapidly assembled into larger composite-services customized to perform specific tasks (shop for a flight, book a flight, shop for a car and hotel, show me pictures of selected offerings, book a flight, car and hotel etc.) for many different products, Business Units and customers.

SUCCESS

- Has your project achieved or exceeded its goals?

Yes. We have substantially increased performance, scalability, stability, overall quality and reuse across every aspect – hardware, software and people. The effort to create a light-weight SOA framework capable of processing composite service efficiently and quickly has been extremely successful. The TDS SOA design was committed to provide greater scalability and stability than the existing framework without degradation in performance.

In 4Q04 we completed the migration of the Galileo Web Services product to the new framework, creating greater scalability and logical levels of granularity as well as superior stability and a 20% improvement in end to end performance. These performance improvements are particularly encouraging as we expect to see further improvements once we upgrade the gateway with XML Appliances.

Other major accomplishments include \$100 million in cost avoidance by moving our Fares system from mainframe to Intel. \$6+ million p.a. cost avoidance through our cookie-cutter-architecture strategy. More streamlined IT lifecycle giving us greater efficiencies through reuse and standardization. A reduction in our production platform footprint – this has resulted in simpler operations and the development of greater core technical competencies. A leading SOA framework etc.

- Is it fully operational?

Yes – We began to implement the strategy in 2002. In 4Q04 we began to take production traffic on our TDS SOA framework showing greater stability and a 20% improvement in performance over the incumbent platform. Also in 4Q04, Galileo became the first GDS in the world to offer fully automated rules running 100% of its fare shopping and pricing on highly redundant, high performing Intel processors.

- How many people benefit from it? If possible include an example of how the project has benefited a specific individual, enterprise or organization. Please include personal quotes from individuals who have directly benefited from your work.

The number of beneficiaries of this project has yet to be determined, but as we begin a migration of our

functionality from proprietary mainframe systems to open systems, the TDS SOA framework will become the target environment for that migration affecting the majority of developers and customers, world-wide, that use access our content.

- How quickly has your target audience of users embraced your innovation? Or how rapidly do you predict they will?

We really had two target audiences of users – the internal development community and end users/customers. Some members of the development community had reservations as to whether the strategies would deliver on their promises – thankfully the initial deliverables (always the most important) have been extremely successful and the overall concepts (greater standardization and reuse) and specific deliverables, TDS SOA framework, NAS, XML Appliances etc. are being employed across Cendant and other organizations we have presented to.

End users/customers probably fall into two categories, the majority being those who don't know (or need to know) about the architectural significance of the process improvements – they just care that it we continue to improve stability, performance and time to market and stability of our e products. We have received several communications from customers expressing appreciation for these improvements.

The other customers are more technically focused and working with them we are beginning to see opportunities that were not envisioned from the off-set. Specific to the TDS SOA implementation, this project was initially targeted to deliver a set of services (shop flight, book flight etc.) through an XML interface/Business Domain Model (TDS Rosetta Stone). However, recent client conversations have exposed a desire for users to not only consume those composite services from outside the framework, but also to deploy their own components inside the framework and consume lower level components (logging, caching etc.), to further extend the efficiencies of their own development.

- Describe future plans for the project

We will continue to extend the reuse strategy while also continuing to find solutions that give use improvements in our TCO - 64 bit, dual core chips. Adding the ESB to our SOA framework, extending the use of appliance solutions and continuous availability databases. All of these will be critical as we develop a robust platform capable of supporting increasing volumes of traffic and disparate content sources.

DIFFICULTY

- What were the most important obstacles that had to be overcome in order to be successful? Technical problems? Resources? Expertise? Organizational problems?

'Turning the ship' was something we knew we needed to do in order for us to be successful. Cendant TDS is a collection of newly acquired businesses all of whom operated under intense pressure and autonomy. Getting those units to see the benefit of collaboration, of sharing and reuse of their commodity assets (even recognizing that they had 'commodity assets') was a major challenge that was only overcome by simultaneously gaining support of senior management and the development staff.

Success in our initial ventures was critical to gaining acceptance, trust and credibility, first beginning with the move to a repeatable hardware configuration (cookie-cutter-architecture) that presented cost avoidance opportunities of over \$6 million in 2003 (through license savings (Linux over Unix) better hardware efficiencies (Intel vs RISC) and improvements in project timelines (through centralizing technical duties)) followed by spectacular cost avoidance savings coming from the migration of our Fares system from mainframe to Intel – estimated at \$100 million over 3 years.

- Often most innovative projects encounter the greatest resistance when they are originally proposed. If you had to fight for approval and/or funding it would be useful to include a summary of the objections you faced and how you overcame them. - Did you encounter any unexpected challenges?

Probably the most controversial decision was to standardize of Java for core business services. While we support any language coded service called externally, provided it presents a workable interface contract, greater efficiencies in terms of performance and reuse increased if the services share a common language and run-time environment. The benefit of Java being that it is built for portability should other (non-X86 for example) technologies present better price for performance opportunities

