CASE STUDY

Travelers Property Casualty Corporation: Building an Object Environment for Greater Competitiveness

Douglas M. LaBoda
Jeanne W. Ross

September 1997

CISR WP No. 301
Sloan WP No. 3975

Center for Information Systems Research
Sloan School of Management
Massachusetts Institute of Technology
77 Massachusetts Avenue, E40-193
Cambridge, MA 02139-4307

CISR Working Paper No. 301

Title: *CASE STUDY—Travelers Property Casualty Corporation: Building an Object Environment for Greater Competitiveness*

Authors: **Douglas M. LaBoda and Jeanne W. Ross**

Date: **September 1997**

Abstract: Stymied by conceptual, technical, organizational, and political challenges, few in-house IT units have been able to capitalize on the proposed benefits of object orientation. The Travelers Property and Casualty Corporation Claim unit has been evolving toward an object environment since 1991, and in the process, has developed three key systems, an object base for future development, and a strong partnership among line managers, Claim unit IT staff, and central IT staff. This case describes how Travelers overcame the challenges associated with implementing an OO environment. The lessons they have learned in this process are relevant to any firm that is rebuilding its IT infrastructure.

24 pages

This paper is also available from the Sloan Working Paper Series as Working Paper No. 3975.

Center for Information Systems Research
Sloan School of Management
Massachusetts Institute of Technology

Travelers Property Casualty Corporation: Building an Object Environment for Greater Competitiveness

Background

On December 1, 1995, The Travelers Property Casualty Company agreed to acquire Aetna's Property and Casualty business. The new Travelers Property Casualty Corporation, a public company and member of The Travelers Group, became the U.S.'s fourth largest property and casualty insurer. Eight days after the announcement of the merger, management of the two insurance units decided that Claim Services—one of three key business units in the new company—should rely on Travelers' existing claim systems and the object-oriented (OO) platforms on which they were built. Nine months after the merger, former Aetna workers' compensation claims had been converted to T-Mate, the Travelers' OO application for processing workers' compensation claims. In the meantime, Claim Services had deployed its second major OO product, IMPACT, a front line workstation for settling Personal Line (automobile, home, and property) claims. By mid-1997, Travelers was rolling out its third major OO system, which had reused most of the components of the IMPACT workstation. Claim Services management was delighting in the capabilities of a platform which, as one manager described it, "has the capacity to do whatever I need it to do." What had started as a slow and risky endeavor into object orientation at The Travelers was paying significant dividends.

This case study describes Travelers Claim Services' journey from the inception of the T-Mate concept in 1991, through the decision to adopt an object-oriented systems development environment, and the firm's eventual reuse of systems components. It describes the obstacles Travelers encountered as it worked its way up a formidable learning curve in the process of building a new infrastructure based on immature technologies. The Claim unit's experience with its OO-based IT infrastructure offers useful insights for any firm that is building OO platforms. The lessons learned also apply to any firm that is developing or enhancing an infrastructure using immature or fast-changing technologies.

Background on Object-Oriented Systems

Under dynamic market conditions, well-designed IT infrastructures can facilitate strategic flexibility and enable faster development of new products and services (Broadbent and Weill, 1997). Object approaches are well suited to fast, flexible infrastructures in part because they create modular, reusable components which can reduce IT development cycle times, leverage development costs over multiple systems, and simplify systems maintenance. Just as important, OO approaches are particularly well-suited for applications supporting knowledge workers. While traditional approaches to system development emphasize the sequencing of activities, OO approaches attempt to embody all the functions in a task by defining the policies and conditions that constrain that task rather than

any particular sequence for performing those functions (Pancake, 1995). Objects contribute to an infrastructure to the extent that they encapsulate functions that will be used or extended in future applications.

Despite a great deal of excitement about the potential for object oriented programming to radically transform software delivery, few firms have established significant OO infrastructures (Fichman and Kemerer, 1993). At least four kinds of obstacles have prevented successful implementations of OO environments in individual firms:

- ***Conceptual difficulties*** — Designing objects rather than sequences of a process requires a fundamental paradigm shift (Adhikari, 1995; Pancake, 1995). While it is reasonably easy to learn how to define an object and even how to hook objects together, it is very difficult to learn how to make smart design decisions. At a minimum, effective design requires both high-level and detailed knowledge of the business process or task that is being modeled. Even where this knowledge exists, the appropriate level of abstraction is difficult to define (Hofman, 1995). Experienced designers typically find that they do not really understand how an object should be structured until they have completed an application, at which time they may need to redo some of the object coding (Pancake, 1995).

- ***Technical difficulties*** — OO adopters must apply new development methods and tools. Many of those tools are immature, which makes them difficult to implement reliably (Berg, Cline and Girou, 1995). The lack of industry standards makes it difficult to know which tools will be supported by vendors in the future, and thus what platforms will prove to be the most robust (Fichman & Kemerer, 1993).

- ***Organizational difficulties*** — Reuse is hard to achieve partially because it is hard for developers to document the precise characteristics of an object or class of objects. Thus, systems designers find it is more difficult to understand existing software components than to create new ones (Karimi, 1990; Pancake, 1995). In addition, programmers are believed to have a "not invented here" bias (Fichman and Kemerer, 1993) which limits their motivation to reuse existing code. Consequently, firms that are focused on reuse find they need structures or procedures that institutionalize reuse (Hofman, 1995).

- ***Political difficulties*** — The development of an OO environment requires substantial time and upfront expenditure before key components are in place to assemble full applications (Berg, Cline and Girou, 1995; Fichman and Kemerer, 1993; Pancake, 1995). Building for reuse can cost 3-10 times as much as developing a single application (Hofman, 1995). Securing funding for shared infrastructures without demonstrable business benefits is, at best, difficult (Weill, 1993).

Travelers Claim Information Systems[1] embarked on a large-scale OO development effort with no prior OO knowledge and limited experience in distributed computing. Staff en-

[1] At the start of the case study, Travelers Insurance had a centralized IT unit with applications development staffs dedicated to each major business unit. Over time, these dedicated staffs were decentralized to the

countered all four challenges described above as they attempted to create not just a system but a platform for continued OO development. This case study is based on over forty interviews conducted between February 1993 and June 1997 with Travelers' IT and business managers, as well as systems developers and users.

Background on Travelers Property Casualty Corporation Claim Services

Travelers Property Casualty Corporation had revenues of $8.2 billion in 1996 on the sale of a full range of insurance products to individuals and businesses. The company had 21,000 employees and sold insurance through 7,200 independent agencies. Travelers competed with large national and international insurance companies like CIGNA, AIG, Kemper and Liberty Mutual, as well as smaller regional firms. By the early 1990s property casualty insurance had become a mature industry suffering from overcapacity. It experienced intense competition on two fronts: price and customer service.

Travelers Property Casualty was comprised of three major business units: Commercial Lines, Personal Lines, and Claim Services. Commercial and Personal Lines represented the firm's two major product lines. Commercial Lines packaged insurance products for businesses, including workers' compensation, property, and liability insurance. Personal Lines focused on the needs of individuals for products like automobile, home, and personal property insurance.

Claim Services was responsible for settling customer claims for three major product lines: workers' compensation, commercial property and liability, and personal lines. The primary opportunities for the Claim unit to enhance the firm's competitiveness were: (1) cutting costs through efficient claims processing; (2) enhancing sales of insurance products by providing improved customer service; and (3) ensuring payouts that were neither too large (which would reduce profits and require increased premiums) or too small (which would fail to meet commitments and lead to customer dissatisfaction).

Because claim processing was a paper intensive process, information technology had always been a critical tool in the industry. In its most obvious role, IT enabled automation that could reduce costs. More importantly, however, Claim Services management believed that information technology would increasingly offer distinctive advantages to firms that could identify and deliver new customer products and services faster than their competitors. To this end, line managers envisioned state-of-the-art applications, while IT managers were mindful of the importance of a solid infrastructure.

Taking the Plunge into OO

In late 1991, a team of branch managers and IT staff submitted specifications for a new workstation to support the Travelers' 1800 workers' compensation claims managers and administrators. The specifications resulted from 18 months of work focused on reengineering the role of the case manager. Nicknamed T-Mate, the workstation was key to the reengineering effort because it would empower case managers through a set of

---

business but they maintained a dotted line relationship to the central IT unit, which was responsible for telecommunications, mainframe processing, and research and development.

automated tools that a skilled and self-sufficient individual could use to classify, investigate, settle, and administer a claim, as well as communicate efficiently with government reporting bodies and customer firms. Goals of the reengineered process included a 10-20% increase in case manager productivity (number of claims handled) and improved customer service (faster response to and settlement of claims; more accurate reporting of insurance costs to large accounts).

Initial prototypes of the system used a graphical user interface product connected to the host system to provide more user-friendly and flexible screens. Comparing one of these prototypes to the existing system in 1991, the team observed that "it was a prettier system but ...[not] a significantly more functional system." The only prototype that aroused the enthusiasm of the design team was one which had a workflow queuing paradigm that required an object oriented development methodology:

> It was like night and day. They [prototype developers] forced us into ten years from now, instead of just five years from now. (T-Mate Design Team Member, 1993)

IT management in Claim Services detailed the known benefits and risks of embarking on development and support of a highly distributed, object-oriented system with the Workers' Compensation Claim management team. Together they agreed to take on the risks of developing a system on immature technologies while building technical expertise in-house.

As staff developed the system, Claim Information Systems worked closely with business partners to establish priorities, train case managers, and define and test functionality. The business vice president championing the project actively monitored progress and interacted regularly with key players. He appointed a full-time line manager responsible for clarifying systems requirements, for organizing training, and for preparing field office staff for process changes.

From an IT perspective, T-Mate was expected to pay for itself in immediate business benefits while also creating a platform for reuse. Because it represented the Travelers' first major foray into not only OO systems but distributed systems, the project took on a very high profile, and top IT management allocated resources to ensure its success. In particular, within corporate IT, a new unit, the Distributed Systems Management team, was charged with creating an "industrial strength" support infrastructure for future distributed systems. As Claim Information Systems was developing Travelers' first large-scale distributed system, the Distributed Systems unit was developing a management process that included electronic, unattended distribution, installation, and configuration of all software; continuous performance monitoring of hardware, system software, and application components; and automated hardware and software asset management.

Confronting OO's Obstacles

The development team encountered conceptual, technical, and political challenges as they worked toward a targeted delivery date of July 1993. Conceptually, they attempted to

generate a large object model describing the elements of the system. While they attempted to incorporate key OO concepts such as encapsulation and inheritance into this model, it was rarely clear how individual objects should be abstracted. Ultimately, they ended up with "this huge object model in which everything depended upon everything else, and it was basically kind of glued together." As a result, the objects needed to be continuously reassessed and revised during development, and the resulting objects "all had a strong Workers' Compensation flavor."

Technical challenges arose from a whole set of new technologies including C++, OS/2, and Microsoft SQL Server, and from a design feature that called for duplicating data from the host on local servers. The distributed data architecture required a complex process of continuously refreshing duplicate data. The immature technologies and ambitious design had a more negative impact on programmer productivity than anticipated. Because vendor products were rapidly improving and no industry standards existed, they had to frequently reassess the tools and systems they had chosen. At one point, the technical developers decided to abandon the compiler they had been using because, as one described it:

> We were having a lot of problems...trying to figure out what was wrong with the application. It was almost like a mystery trying to correct problems. If you were able to correct it, it was luck. It wasn't a science. (T-Mate Technical Director, 1993)

The technical developers, in conjunction with the Distributed Systems Management Team, also had to reexamine the choice of OS/2 as the LAN operating system when two large offices outgrew OS/2 before it was implemented. In that case, they decided the change would be too disruptive, and they opted to make the two large offices nonstandard.

Politically, IT management attempted to articulate both the costs and the benefits of building for reuse. While development of the support infrastructure was centrally funded, the object modeling and coding were funded by the Workers' Compensation business as part of the T-Mate project. Other business units were not interested in funding an initiative of unknown future value. The Workers' Compensation unit was happy to have its systems reused in the company, but managers did not want reuse considerations to delay T-Mate's delivery. Thus, reuse was funded only to the extent that it grew out of T-Mate development:

> There's a very strong hesitancy in the corporation about this whole concept of reuse. It's very much an uphill battle. It is funded, but it's continually being challenged and checked. That's because the real benefit to the corporation is, obviously, supporting the business and coming up with the right reengineering strategy to make us more productive. (T-Mate Technical Director, 1994)

As they tackled these challenges, the development team started to miss interim milestones. They found it necessary to relax concerns about reusability in order to meet immediate business needs and deliver something of value to the business. Team members worked with business managers to identify functions that could wait until later versions in order to reduce the scope of the project. Always apprised of the setbacks, business unit management continued to prepare branch office personnel for system and process changes.

T-Mate Implementation

In November 1993, six months late but on budget because of a dramatic fall in hardware prices, T-Mate was piloted in three branch offices. Users at the pilot offices praised the system's design and functionality, but the technology was unstable. System crashes were common, response time was often slow, and occasional mismatches between mainframe and local data required awkward work-arounds. But the field office staff communicated daily with the development team and they received weekly enhancements that slowly increased stability.

Even as IT staff attacked the technical problems exposed by the pilots, the business champion argued for nation-wide roll-out of the system:

> You can't take the chance with something this good, not to get it out there. As long as everybody is willing to understand the down side and the up side, that's the way we'll go. (Vice President, Workers' Compensation Claim, 1994)

Reluctant to roll out unstable technology but mindful of the vice president's very significant reengineering task, IT management agreed to move ahead starting in January 1994. By July T-Mate had been installed on 1800 desktops and servers had been installed in all 50 branch offices.

Immediately following the roll-out, the system averaged as many as 250 individual workstation crashes per day[2] and continued to struggle with slow response time. Around 20% of Claim managers opted to ignore the new system during this period and to process their claims on the old host system. Others periodically abandoned T-Mate when they encountered technical difficulties. But the offices that were experiencing the most persistent problems objected to the suggestion that they shut down T-Mate until the technical problems were fixed. Similarly, case managers who were without T-Mate, such as employees who worked at home or at satellite offices whose remote connections were too slow to use, kept begging for the system. As one branch manager explained it:

[2] Two hundred fifty workstation crashes translates to approximately one out of eight workstations crashing once a day. In most cases a crash consumed approximately 20-30 minutes of a case manager's time as the rebooting involved reloading the client software from the server.

> When a tool like T-Mate comes into play, it makes your life easier even though there are so many ... problems with it. The good outweighs any of the bad. (Branch Manager, 1997)

In June 1994, the IT director identified four key problems that appeared to be at the root of the instability. He assigned a team to each problem and empowered each team to take necessary actions to resolve the problem. In one case, a team hired a technical expert to help them work through a specific function that was causing crashes. In another case, the team spent more than a month working intensely with a vendor to isolate the cause of response time problems in a couple of the largest offices.

Code changes and hardware enhancements eventually addressed the technical problems, and in early 1995 all but the two largest offices had stabilized (problems there were resolved several months later). Crashes dropped to approximately one workstation crash per office per day, which, given the state of the technology, was considered good. The number of transactions processed on T-Mate climbed steadily as business management rallied the claim staff to transition from old work processes to new. The addition of major new functionality in both 1995 and 1996 allowed as much as 90% of Workers' Compensation claim transactions to be completed on T-Mate[3]. When T-Mate was rolled out to former Aetna Claim offices in January-April 1997, the implementation proceeded flawlessly, and the case managers enthusiastically embraced the workstation and the new processes, which they described as far better than what they had before. In summarizing the transition, one member of the implementation team said:

> It was a slam dunk. All the prior aggravation paid off. (T-Mate Project Manager, 1997)

Behind the scenes, the infrastructure team provided centralized support of the distributed systems. They ran quality assurance tests and then downloaded system enhancements overnight to servers in each of the branch offices. As case managers turned on their machines in the morning, they automatically received the upgraded software from the server with a message about changes that had been made. Between July 1994 and June 1997 the central unit downloaded major new production versions of the software every three months and also delivered almost 100 smaller enhancements as needed to resolve technical problems, to add functionality, and to adjust features. The infrastructure unit also monitored the systems through probes inserted into the application that allowed staff to identify and fix problems rapidly.

The infrastructure unit met weekly with application developers and felt responsible for the success of T-Mate. One member of the project team noted that central staff were very responsive to concerns about performance after roll-out:

---

[3] Some financial transactions and the assignment of claims that were transferred from other insurance companies were not incorporated into T-Mate and line management did not believe such changes would be worth the effort.

> We had gone into production in the very first office, and they were having performance problems, so I said, "Let's have a war room and we'll get people together," and [the infrastructure support manager] said, "I want you to stay out of it. You're in the business, you don't have to deal with this stuff. It's my job. I remember thinking, "What? You're telling me to stay out of it?" But he handled it. That's basically how it works today. The organizations respect each other, and people handle their jobs. It's as simple as that. (Technical Project Director, 1997)

Reusing OO Components

In February, 1994, while T-Mate was still in roll-out, the Claim information systems unit started developing IMPACT, a workstation to support Personal Lines claims. While pleased with the business outcomes of T-Mate, the development team was disappointed with the OO model they had developed. By failing to isolate business components from data and technology components, T-Mate had few objects that could be reused in the firm's second OO system. Some concepts, such as desktop icons, diaries, and navigation among separate components, were reusable, but in all cases code changes were necessary to make them more generic.

As they prepared to launch the IMPACT development effort, a small group of developers started conceptualizing a three-tier object model. At the top of the three-tier architecture were the business slices, the objects which encapsulated the business components. These objects had parallel objects in the second tier, which was referred to as the Frameworks. The objects in the second tier provided the logic for translating the business components for purposes of accessing systems resources, such as databases. The Frameworks acted as an object request broker that allowed the queuing of requests so that systems could scale up and handle large numbers of requests. The bottom tier consisted of platform objects that interfaced with specific technical platforms.

This three-tier model allowed the development team to isolate business application development from technical platform development, which meant that individual programmers could code a business object without knowing what specific technical platform it would run on. It allowed the Claim IT unit to create small teams specializing in each layer, so that team members jointly designed the objects created by their team but needed only limited understanding of the objects created by other teams. Specifically, they needed to know what functionality the other team was building and what information needed to pass between the two. This paved the way for component reuse.

By May 1995, after fifteen months in development, IMPACT had been delivered to 150 Personal Lines claim handlers. Although IMPACT could not reuse T-Mate code, it did reuse much of the IT staff's new technical skills, the centrally-provided support infrastructure, and the learning Claim Information Systems staff had acquired as to how to work with business partners and build OO systems. Further development was delayed temporarily while the Aetna business was absorbed, but the system was deployed on an-

other 150 desktops between November 1996 and February 1997. Following some organizational changes in the field, it would eventually reach 1700 claim handlers nation-wide.

The three-tier OO architecture was put to the test when a commercial property and liability (CPL) workstation entered the development stage in August 1996. The design allowed for full reuse of the bottom two layers of the IMPACT system. It also reused better than half of IMPACT's business objects (top tier). The CPL workstation was rolled out to five locations in June 1997 and scheduled to be on 600 desktops by October. All 1800 claim handlers were scheduled to have the new system by mid-1998. Along with T-Mate and IMPACT, CPL was positioned as a cornerstone system for Travelers' Claim Services.

A key to reuse within Claim Services was the IT unit's approach to leveraging technical expertise. Travelers hired ten consultants for early stages of T-Mate development. Because their limited understanding of the business seriously impeded their effectiveness, all but one consultant was phased out by the end of 1993. For the most part, Claim Information Systems developed OO expertise internally. Some of the training involved formal classes, but most of the learning was hands on. The unit both leveraged and grew this learning by staffing new development efforts with individuals who had worked on prior OO systems. For example, IMPACT and CPL team members had experience with T-Mate. Their knowledge of its strengths and limitations was critical in designing the next generation of OO systems.

Retention among Travelers' approximately two dozen OO experts was 80-90%. Although some might have earned more elsewhere, Claim Information Systems staff indicated that they chose to stay at Travelers sometimes for personal and family reasons but mostly because they liked the work environment.

Deriving Value from the OO Environment

The most easily measured benefit of the OO environment at Travelers was its impact on cost and development time for new systems. While the Workers' Compensation System cost more than $7 million and took 80 person-years to develop over the course of 24 months, IMPACT cost $1.5 million and took 18 person-years to construct in 15 months elapsed time, and CPL was developed for less than $.5 million and took 6 person-years in 10 months elapsed time. The next system which would be developed was expected to continue the downward trend in development cost and time.[4]

The more important benefits of the OO environment were the business benefits, those already generated by process changes instituted with the new systems and those future changes that the object environment had the flexibility to enable. T-Mate had generated some productivity improvements[5], but management felt that the value of the system lay in a higher quality work product and in improved customer service. An executive vice president noted that the T-Mate platform offered the potential for significant organizational change:

> T-Mate gives us and the systems that are behind it the ability to create reliable, accurate measurements against objectives. People have a much clearer idea of what it is they're expected to do. We then tie a compensation system to that, composed of both salary adjustment and incentives built around performance against those objectives. You start to have a pretty powerful organization. (Executive Vice President, Claim Services, 1997)

In addition to organizational changes, T-Mate provided opportunities to develop new product offerings. This potential was tapped in the development of TravComp 2000, a Workers' Compensation claim innovation that integrated claim processing and medical management, thereby enabling a whole new approach to workers' compensation. As soon as a claim was filed, the system assigned it to Claim Services' staff according to needed experience and special skills. By assigning nurses and diagnosing medical care needs according to the nature of an injury, TravComp 2000 was expected to improve customer service, reduce lost time, and save a great deal of expense dollars for both Travelers and self-insured customers. When it was rolled out in June 1997, Travelers management believed that none of its competitors had the technology to offer a comparable level of service. Traveler's integration of claim handling and medical management technologies and information presented a unique revenue-generating service opportunity.

---

[4] As of June 1997 Travelers had modeled its next two application systems, but for competitive reasons they are not described here.

[5] Industry changes and the Aetna merger had resulted in a new mix of policies that made it difficult to compare pre- and post-T-Mate case manager productivity based on number of claims handled. Claim managers estimated productivity increases of 2-10%. More noticeably, planned field office reorganization would expand managerial span of control and claim handlers could address more of the customers' needs.

The outcomes of the IMPACT implementation were more easily measured. Within the first year of operation IMPACT resulted in a 15% reduction in claim handling costs, mostly through reduced personnel. This cost reduction resulted largely from new functionality that allowed a single claim handler to process a claim from report to resolution without any hand-offs. The CPL workstation was expected to lead to the same kind of efficiencies as IMPACT.

A final benefit of the OO environment at Travelers was the high reliability and low cost of operating and supporting the systems as a result of the decision to centrally control distributed servers from headquarters. When Travelers merged with Aetna, Claim Information Systems found that Aetna's systems administration costs ran 60% higher than Travelers. The Distributed Systems Management team within the central IT unit had met its initial objective in 1994 to staff just one operations technician for every 20 servers. By 1997 the ratio was up to 100 servers per technician.

Tackling the OO Obstacles

Travelers achieved the benefits described in the prior section because it was able to overcome the obstacles that confound most organizations attempting to build OO environments. Travelers' strategies for addressing the four challenges that have frequently stymied the growth of object approaches in in-house systems units are summarized in Table 1 and described below.

Conceptual difficulty — Claim Information Systems dealt with conceptual complexity by accepting the reality of the need for iterative design, clarifying objects through use, and reverse engineering where appropriate. Consistent with the literature (Pancake, 1995), Travelers found that trying to build a detailed object model prior to development was overwhelmingly complex. Modeling and coding objects proved to be a highly iterative process. The more objects they developed, the better they understood how the objects should be modeled. For example, in developing CPL, the team identified components in IMPACT that reflected the Personal Lines business but would have been more useful as generic concepts. The CPL project manager explained:

> They [objects in IMPACT] were written around the domain expertise that we had at the time. If we'd tried to write for everything the first time, we'd have written nothing. That's why it's more important to understand the business than the technology when you're doing objects. The artsy part of the design was figuring out at what level inheritance stops making sense. Have I oversimplified this class to the point where it could be used for anything? (Project Manager, 1997)

Changes like this were becoming easier, and developers expected that the process of better conceptualizing objects was an ongoing one. Some credited the ambitious scope of the first OO development effort (T-Mate) with accelerating the learning process and leading to a solid model that provided the extensive reusability in the second effort.

Technical difficulty — Travelers adopted three strategies to cope with fast-changing and immature technologies. First, the decision to develop OO expertise among some of the firm's most talented IT professionals and then entrust them with the necessary resources and autonomy to resolve perplexing technology problems prevented the technologies from becoming an overwhelming obstacle. Second, the centralized support infrastructure around OO implementation proved critical to responding to problems early in the T-Mate installation, to making all three systems affordable to the firm, and to relieving developers of concerns regarding ongoing support. Third, the three-tier architecture allowed individual staff members to specialize on a limited range of objects and thus simplified the technical demands on individual programmers.

Organizational difficulty — By establishing the scope of the OO environment as Claim Services only (a large division with an IT staff of around 250 persons), Travelers had a unit that was large enough to generate value from reuse but small enough to rely on a team of manageable size (approximately two dozen OO professionals). The three-tier architecture was helpful in supporting reuse because OO staff specialized in different layers of the architecture. Particularly at the lower two layers, specialists designed their objects with fairly complete knowledge of how those objects might be used in the future. The IT unit then tackled reuse by assigning people knowledgeable in prior applications to subsequent development projects, and project managers designed systems with the intention of reusing objects.

Political difficulty — Claim Information Systems management accepted that business managers would tend to reject funding requests for building objects for reuse. Accordingly, the OO environment was built on three key application systems that focused on immediate business needs, sacrificing reuse when the cost in either time or money was substantially higher than it would otherwise be. In particular, T-Mate was not rewritten when developers recognized the limitations of its design. Instead, components were gradually changed as common objects were developed for future systems. IT managers communicated regularly with business managers about the value of reusable objects and won commitment for the concept, but developers were conscientious about need to keep costs low:

> I've made it my job to deliver the functions as cheap and as best I can. If I get into whose budget is funding or who's paying for this piece of the code, we'll never win. I think part of it is the relationship with the business. As long as I'm meeting their goals, they're not worried about running my job. (Project Manager, 1997)

Table 2 summarizes the OO experience at Travelers. The three-tier architecture and specialized small team organization structure that Travelers successfully deployed in 1996 evolved from much trial and error. In 1992 conceptualization of the objects was erratic, the project team of 24 developers was unmanageable, and the technologies were flawed. The Claim unit's success in delivering a large-scale project resulted primarily from the competence of the key players, management persistence in sticking with the project when

it was floundering, intense focus applied to delivering business value and solving technical problems, and a strong partnership among workers' compensation business management, the Claim Services IT unit, and the central IT unit. This success was important to the business because of the capabilities it provided and the learning it delivered. Learning from both what went right and what went wrong in its first OO undertaking enabled the development team to establish formal processes for defining objects in subsequent systems.

Lessons Learned

Table 3 lists six key lessons that Travelers has learned through its efforts to develop object-oriented systems and infrastructure. These are explained below:

1. Recognize that OO development in the pure sense is not practically implementable. Business needs change rapidly and cycle times are short, so the time required to fully conceptualize the OO model will never be available. Consequently, the really elegant OO solution will be elusive. Claim Information Systems has found that building an OO infrastructure is an evolutionary process. The objects kept improving with each new system.

2. Accept that technology change is a constant that will disrupt delivery, implementation, and operations. Claim Information Systems recognized that technologies were unstable, but estimates of development time had not allowed for changes in tools and systems in the midst of development. Once in place, they needed to port systems to new operating environments to take advantage of improved reliability, faster response times, and changing industry standards. IT management at Travelers started to allow additional development time for delays caused by technology changes, and it emphasized the need to have utility objects that allow them to convert systems from one technology platform to another without changes in the business objects.

3. Expect some pain but provide rapid relief. Part of managing expectations in an OO environment is to make clear that new systems and immature technologies require that adopters simultaneously adjust to technical problems and organizational change. This will inevitably lead to pain or discomfort on the part of adopters. The business champion at Travelers played a critical role in preparing individual employees for what to expect. Recognizing that most employees would endure pain for some finite period of time before giving up on the project, IT focused on resolving problems as quickly as possible.

4. Adopt a learning attitude that recognizes the need to learn from mistakes. Managers at Travelers disagree as to whether T-Mate was rolled out prematurely. Some believe that a delayed roll-out would have prevented some of the pain. Others feel that the pain created useful pressure on both the IT and the business organizations to move quickly to make the system workable, so that the firm could start realizing the benefits that much sooner. They all agree that an environment as complex as the one they created necessarily involved trial and error. Viewing mistakes as important to the

learning process allowed organizational members to disagree with one another, to seek out and expose errors, and to invite conflict.

5. Design a reliable, cost-effective architecture for supporting the OO environment. The centralized management of Travelers' distributed servers was essential to reliable and cost effective support. It allowed Claim Services to distribute system revisions to all locations as soon as they were tested and approved. In addition, the centralized support architecture allowed Claim Information Systems to focus on application requirements confident that support needs would be addressed by experts in that function.

6. Hand over ownership for technology problems. No one project manager or department director can understand all the technologies and techniques required to build and maintain a distributed environment. Travelers' management recognized that, in order to succeed, everyone involved in the system and its support had to deliver. This required a great deal of trust among Claim Information Systems, central IT, and Claim Services line management. It meant that both IT and line management had to clearly scope and hand over accountability for each problem, provide resources necessary for addressing the problem, and then let staff solve the problem as they saw fit.

Although the lessons learned refer specifically to the development of object oriented systems and infrastructures, they pertain to the development of any IT environment that relies on immature or fast-changing technologies. Large systems and infrastructure components are costly and time-consuming to develop (Ross, Beath and Goodhue, 1996). Consequently, business demands and technologies will inevitably change during the development life cycle. Even organizations that attempt to overhaul legacy systems by purchasing large-scale enterprise resource planning (ERP) systems find that technologies change and configurations get altered during the implementation process. What Travelers demonstrates is that a well-conceived plan and consistent attention to immediate business needs allows a firm to evolve toward a solid platform of systems and capabilities.

For most organizations, technology change is a constant that will cause disruptions in IT and business processes, and this will cause some pain. IT and business unit managers must recognize their joint responsibility for making each system and process implementation a successful learning experience. Business partners must buy into the discomfort that implementations will cause and focus attention on the business benefits to be gained. IT units need the help of business partners in identifying the individuals and operating units that have the creativity, open-mindedness, and persistence to participate in the early learning stages of new technology introductions. For their part, IT units must demonstrate the competence and sensitivity to respond to problems.

In this case study, it is difficult to distinguish between applications and infrastructure, because existing application components become the foundation for new application development. The quality of the infrastructure emerges from the quality and strategic value of the individual systems that constitute its key components. Increasingly, the flexibility of a firm's IT infrastructure and IT applications and the alignment of IT with strategic objec-

tives define a firm's ability to compete in the marketplace. This technology base cannot be dropped into place. It must be pursued through thoughtful planning and relentless determination.

September, 1997

References

R. Adhikari, "Adopting OO Languages? Check Your Mindset at the Door," *Software Magazine,* November 1995, pp. 48–59.

W. Berg, M. Cline, and M. Girou, "Lessons Learned from the OS/400 OO Project," *Communications of the ACM* (38:10), October 1995, pp. 54–64.

M. Broadbent and P. Weill, "Management by Maxim: How Business and IT Managers Can Create IT Infrastructures," *Sloan Management Review* (38:3), Spring 1997, pp. 77–92.

M.E. Fayad and W. Tsai, "Object-Oriented Experiences," *Communications of the ACM* (38:10), October 1995, pp. 51–53.

R. G. Fichman and C. F. Kemerer, "Adoption of Software Engineering Process Innovations: The Case of Object Orientation," Sloan Management Review (34:2), Winter 1993, pp. 7–22.

Hofman, D. "Radical Change in the Systems Delivery Process: Object-Oriented Component Assembly," CISR WP No. 280, June 1995.

Karimi, J. "An Asset-Based Systems Development Approach to Software Reusability," MIS Quarterly (14:2), June 1990, pp. 179–198.

Pancake, C.M. "The Promise and the Cost of Object Technology: A Five-Year Forecast," Communications of the ACM (38:10), October 1995, pp. 33–49.

Ross, J.W., Vitale, M.R. and Beath, C.M. "The Untapped Potential of IT Chargeback," CISR WP No. 300, June 1997.

Weill, P. "The Role and Value of Information Technology Infrastructure: Some Empirical Observations," in Banker, R.D., Kauffman, R.J. and Mahmood, M.A., *Strategic Information Technology Management,* Harrisburg, Pennsylvania: Idea Group Publishing, 1993.

Table 1
Strategies for Overcoming OO Obstacles at Travelers' Claim Services

Obstacle	Strategies
Conceptual: Design of object model	1. Pursue modeling and coding as an iterative process 2. Select large core system as first project
Technical: Immaturity and instability of OO technologies	1. Develop in-house expertise and allow experts to solve problems 2. Create centralized infrastructure support 3. Develop three-tier architecture to permit developers to specialize on a limited number of technologies
Organizational: Enforcing reuse	1. Reuse expertise 2. Use three-tier architecture to develop specialized understanding of how objects will be used
Political: Securing funds for infrastructure components	1. Make low cost and immediate business value top priorities 2. Communicate constantly with business partners

Table 2
The Evolution of Objects and Reuse within Travelers Claim Systems

	1992	1993	1994	1995	1996
• **Object Design/ Methodology**	• No formal standards • Use of Booch notation for design diagrams	• Attempt at formalizing the business design process	• Renewed effort to redefine a formal process of design apps. (Use of walkthrus, checklists, and standard output documents)	• Home-grown adaptation of various OO methodologies, based on the new frameworks infrastructure	• Some evolution of home-grown methodology
• **Architecture**	• Use a set of foundation classes (home-grown)	• Continued refinement of home-grown foundation classes • Use of a single data persistence "engine" for instantiating objects • No use of independent components	• Renewed emphasis on architecture • A client/server messaged-based transaction model emerged: "the frameworks" • Emphasis on isolated component development	• The frameworks completed (written concurrently with the 2nd app.) • Use of client/server model, but run both client and server on the same "client" device	• Frameworks-based
• **Organization**	• 1 large OO development team • 24 developers, including 10 consultants • 2 project managers, 1 senior OO consultant, 1 client/server architect	• 2 OO development teams (a small team to do foundation work first) • 25 developers • 1 project manager, 1 senior OO consultant, 1 client/server architect	• 3 small development teams (1 to maintain the first app., 1 to begin design work on the 2nd app., and 1 to manage the foundation classes and infrastructure) • 1 consultant left	• 3 small development teams (2 business, 1 infrastructure/ technical)	• 5 very small development teams with overlap of responsibilities (3 business app. oriented, 1 business reuse oriented, 1 infrastructure/technical)
• **Project Management**	• Failed attempts to use traditional project management approaches	• Identified and completed key "global" components • Reporting of project milestones using % complete approach	• Transitioned back to traditional project management techniques	• Successful use of traditional project management techniques with new component-based architecture	• Traditional project management

	1992	1993	1994	1995	1996
• **Philosophy**	• "Make the dates" • "Reuse is secondary"	• "Deliver something to show progress" • "Add staff to make dates" • "Reuse is secondary"	• "Reuse as much of the 1st app. As possible" • "Reduce complexity"	• "Strive for quality to maximize stability" • "Prepare for more apps."	• "Enforce specialization of existing components for reuse" • "Add additional frameworks functionality to support all apps."
• **System Software**	• OS/2.13 • LAN Server 3.0 • OS/2 EE • MS SQL Server • MVS/CICS/DB2/ IMS	• OS/2 2.11 • LAN Server 3.0 • OS/2 Comm. Mgr. • MSSQL Server • MVS/CICS/DB2/ IMS	• OS/2 2.11 • LAN Server 3.0 • OS/2 Comm. Mgr. • MS SQL Server • MVS/CICS/DB2/ IMS	• OS/2 2.11 • LAN Server 3.0 • OS/2 Comm. Mgr. MS SQL Server • MVS/CICS/DB2/ IMS	• OS/2 2.11 NT • LAN Server 3.0, NT Server • OS/2 Comm. Mgr., Attachmate for NT • MS SQL Server, Sybase • MVS/CICS/DB2/IMS
• **Toolset**	• Zortech C++ • "Hybrid" C++/Views • PVCS	• Borland C++ • "Hybrid C++/Views • PVCS	• Borland C++ • C++/Views (1st app.) • zApp (2nd app.) • PVCS	• Borland C++ • C++/Views (1st app.) • zApp (2nd app.) • PVCS	• Borland C++ • C++/Views (1st app.) • zApp (2nd, 3rd, 4th apps.) • PVCS
• **Time to Implementation**		• 24 months to full 1st app.! (80 person years)		• 15 months to 2nd app. (18 person years)	• 10 months for 3rd app. (6 person years) (Completion date: 6/97)
• **Development Cost**		• $7 million		• $1.5 million	• $0.5 million
• **Number of Production Users**	• 0	• 140	• 1800	• 1800 (1st app.) • 150 (2nd app.)	• 2500 (1st app.) • 1700 (2nd app.) • 1800 targeted (3rd app.), first 600 in 1997

Prepared by Ron Calabrese, Technical Director, Claim Information Systems

Table 3
Key Lessons from the Travelers' OO Experience

- Recognize that OO development in the pure sense is not practically implementable.
- Accept that technology change is a constant that will disrupt delivery, implementation, and operations.
- Expect some pain but provide rapid relief.
- Adopt a learning attitude that recognizes the need to learn from mistakes.
- Design a reliable, cost-effective architecture for supporting the OO environment.
- Hand over ownership for technology problems.