With SOA, corporate geeks finally get to be part of the business adventure again, developing new ways to use technology to grow the firm, helping to spot new trends and opportunities, and seeing new ideas to fruition. But before you go marching off to save the world, we have some more explaining to do. A story will help.

Once Upon a Time

Once upon a time, there was an insurance company called ABC Insurance Incorporated. When ABC was born, oh, maybe 150 years ago, it began by selling insurance policies to factories and manufacturers. In those days, there were no computers to mess things up. A nice person sent a letter inquiring about a policy. A smart person set a rate, sold a policy, and hoped that nothing caught fire or blew up. ABC thrived for more than a hundred years.

But then, things got complicated. Other companies started stealing their business. Customers were asking for insurance for different kinds of risk. ABC had to change or die.

ABC was an early user of punch-card accounting systems. In the 1960s, ABC bought computers and hired programmers and built software applications to support its business. In the 1980s, it bought software packages from different suppliers to help it continue to compete. It bought or built business applications to solve problems all over the company — one at a time. For example, it bought an application for the corporate finance department, created one to handle customer claims, and procured other applications to manage research information about what type of accidents were most common under what circumstances.

This worked well for many years, until the 1990s, when ABC found itself competing against financial services companies who decided *they* could sell insurance, too. Suddenly, ABC needed to find new ways to make money that didn't cost too much. Its leaders thought up exciting new solutions based on the knowledge of their business and their customers and through new, cool *technology.*

In addition, Management thought ABC could better compete by acquiring some other insurance companies with complementary products. ABC could sell these new products to existing ABC customers and sell ABC's products to the customers of the companies they acquired. These smart guys and gals understood business strategy. Everyone got really excited until . . .

Management talked to IT, and IT said, "This is really, really exciting, but we have a *small* problem."

“What could it be?” cried Management.

“It is this,” said IT. “We can no longer simply buy or build more programs to implement our new moneymaking, cost-saving ideas. Everything we want to do has to work in concert with what we already have. The very running of our company depends on all the business applications that we have built and acquired over years working together smoothly — the programs to tally the money coming in, the programs to administer the claims processing going out, to do risk analysis, premium billing, payroll, invoicing, and sales commission calculation. When you come right down to it, our company is the aggregation of all our programs. Everything we need in order to carry out our day-to-day business functions — all our policies and information, including all the information about our customers — is locked inside these programs.”

“Well,” said Management, “You can just write new programs to tie everything together. We’ll *integrate* and we will all be very happy.”

And IT said, “Yes, it is possible to *integrate*, but integrating will take *a very, very long time.* Integrating will take at least 18 months, maybe 2 years, and by then you may want more changes that will take another 18 months or 2 years, and by then it may be too late. And,” IT continued, “*it will cost lots and lots of money.*”

Management and IT were very sad. They knew that ABC would not survive if they couldn’t find *a new way of thinking.* So they began asking everyone they knew if there was any way to save ABC. They searched and they studied and they prayed until one day a package arrived from Amazon.com. In that package were several copies of a yellow-and-black book. On the cover of the yellow-and-black books, they read *Service Oriented Architecture For Dummies.*

Both Management and IT took copies of the book and read. They were very excited to discover that they didn’t have to throw stuff away and that they could reap benefits in a short time. In the end, they came up with a *new* strategy, one based on four key elements:

1. The IT organization will partner with the line of business managers to create a high-level map of what the business will look like.
2. The IT organization will create a flexible structure that will turn key IT software assets into reusable services that can be used no matter how the business changes. These services will include everything from business processes and best practices to consistent data definitions to code that performs specific business functions.
3. The IT organization will use only accepted industry standards to link these software assets together.

4. The IT organization will use the service oriented architecture concept described in the rest of this book to begin to create business services that are consistent with the way the business operates.

Together, Management and IT began a journey, and, as far as we know, they are living happily ever after . . . In Part V, we give you many real-life case studies from real-life companies you may know that indeed are alive and well and living happily on their *Journey to SOA.*

Better Living through Reuse

One of the biggest deals in the SOA world is the idea that you don't throw things out. You take the stuff (software assets) that you use every day — well, the *best* of the stuff you use every day — and package it in a way that lets you use it, reuse it, and keep on reusing it.

One problem common to many large companies that have been around for a while is that they have lots of similar programs. Every time a department wants something slightly different, the department builds its own version of that something so that, across a particular company, you can find umpteen versions of more or less the same program — with, of course, slight variations. Many IT shops have policies and procedures designed to prevent this sort of thing, but when deadlines loom and budgets are tight, it's often easier and quicker to write something from scratch that fills the need rather than coordinate with other divisions. This sort of duplication also happens a lot when one company acquires another and finds that they have similar (but not identical) programs purporting to do the same thing.

These slight variations are precisely what make systems very complicated and expensive to maintain — if you make a change in business policy that affects the sundry applications, for example, you have to find and change each and every instance in every application that is affected. And even the slightest difference in implementation can result in inconsistencies — not a nice thing to find when those compliance auditors come snooping.

With SOA, these important programs become *business services.* (We talk more about this in Chapter 2.) You end up with one single business service for a given function that gets used everywhere in your organization. With SOA, when you need to change a business policy, you change it in one place and, because the same service is used everywhere, you have consistency throughout your organization.