how an application went sideways as "Whac-A-Mole." When things go wrong in their heavily SOA-based environment, it's hard to determine what happened, because twenty-plus systems are involved in the processing of a certain transaction, making it really hard for the IT department to track down exactly why and where things went wrong. (We've all seen this movie: Everyone runs around the war room saying, "I didn't do it!"—there's also a scene in that movie where everyone is pointing their fingers at you.) We helped this client leverage a Big Data platform to analyze approximately 1TB of log data each day, with less than 5 minutes latency. Today, the client is able to decipher exactly what is happening across the entire stack with each and every transaction. When one of their customer's transactions, spawned from their mobile or Internet banking platforms goes wrong, they are able to tell exactly where and what component contributed to the problem. Of course, as you can imagine, this saves them a heck of a lot of time with problem resolution, without imposing additional monitoring inline with the transaction, because they are using the data exhaust that is already being generated as the source of analysis. But there's more to this use case than detecting problems: they are able to start to develop a base (or corpus) of knowledge so that they can better anticipate and understand the interaction between failures, their service bureau can generate best-practice remediation steps in the event of a specific problem, or better yet they can retune the infrastructure to eliminate them. This is about discoverable preventative maintenance, and that's potentially even more impactful.

Some of our large insurance and retail clients need to know the answers to such questions as, "What are the precursors to failures?", "How are these systems all related?", and more. You can start to see a cross-industry pattern here, can't you? These are the types of questions that conventional monitoring doesn't answer; a Big Data platform finally offers the opportunity to get some new and better insights into the problems at hand.

The Fraud Detection Pattern

Fraud detection comes up a lot in the financial services vertical, but if you look around, you'll find it in any sort of claims- or transaction-based environment (online auctions, insurance claims, underwriting entities, and so on). Pretty much anywhere some sort of financial transaction is involved presents a potential for misuse and the ubiquitous specter of fraud. If you

leverage a Big Data platform, you have the opportunity to do more than you've ever done before to identify it or, better yet, stop it.

Several challenges in the fraud detection pattern are directly attributable to solely utilizing conventional technologies. The most common, and recurring, theme you will see across all Big Data patterns is limits on what can be stored as well as available compute resources to process your intentions. Without Big Data technologies, these factors limit what can be modeled. Less data equals constrained modeling. What's more, highly dynamic environments commonly have cyclical fraud patterns that come and go in hours, days, or weeks. If the data used to identify or bolster new fraud detection models isn't available with low latency, by the time you discover these new patterns, it's too late and some damage has already been done.

Traditionally, in fraud cases, samples and models are used to identify customers that characterize a certain kind of profile. The problem with this approach (and this is a trend that you're going to see in a lot of these use cases) is that although it works, you're profiling a segment and not the granularity at an individual transaction or person level. Quite simply, making a forecast based on a segment is good, but making a decision based upon the actual particulars of an individual transaction is obviously better. To do this, you need to work up a larger set of data than is conventionally possible in the traditional approach. In our customer experiences, we estimate that only 20 percent (or maybe less) of the available information that could be useful for fraud modeling is actually being used. The traditional approach is shown in Figure 2-1.

You're likely wondering, "Isn't the answer simply to load that other 80 percent of the data into your traditional analytic warehouse?" Go ahead and ask your CIO for the CAPEX and OPEX approvals to do this: you're going to realize quickly that it's too expensive of a proposition. You're likely thinking it will pay for itself with better fraud detection models, and although that's indeed the end goal, how can you be sure this newly loaded, cleansed, documented, and governed data was valuable in the first place (before the money is spent)? And therein is the point: You can use BigInsights to provide an elastic and cost-effective repository to establish what of the remaining 80 percent of the information is useful for fraud modeling, and then feed newly discovered high-value information back into the fraud model (it's the whole baseball left hand–right hand thing we referenced earlier in this chapter) as shown in Figure 2-2.

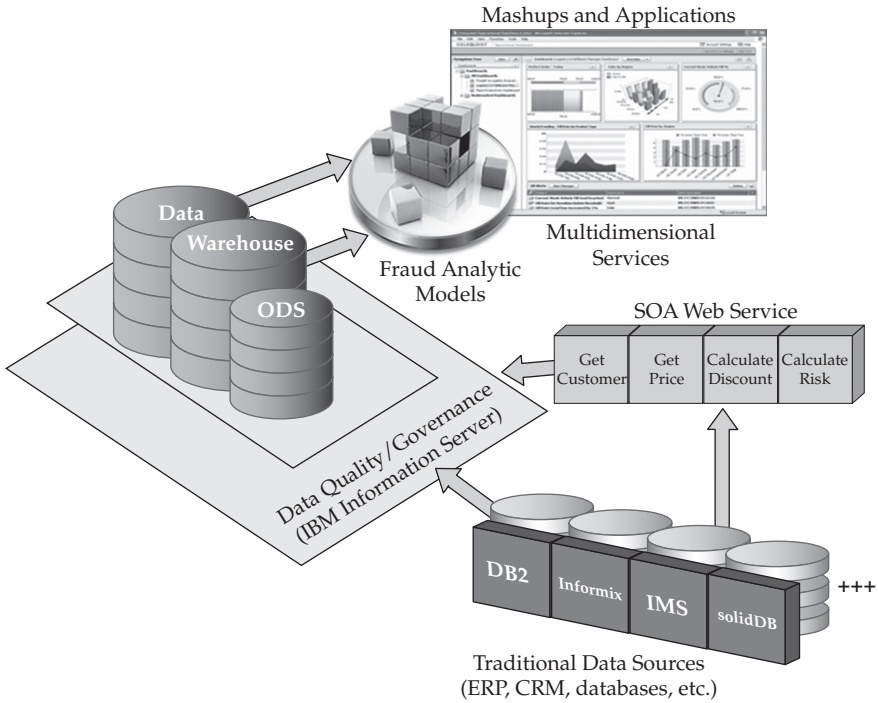

Figure 2-1 *Traditional fraud detection patterns use approximately 20 percent of available data.*

You can see in Figure 2-2 a modern-day fraud detection ecosystem that provides a low-cost Big Data platform for exploratory modeling and discovery. Notice how this data can be leveraged by traditional systems either directly or through integration into existing data quality and governance protocols. Notice the addition of InfoSphere Streams (the circle by the DB2 database cylinder) as well, which showcases the unique Big Data platform that only IBM can deliver: it's an ecosystem that provides analytics for data-in-motion and data-at-rest.

We teamed with a large credit card issuer to work on a permutation of Figure 2-2, and they quickly discovered that they could not only improve just how quickly they were able to speed up the build and refresh of their fraud detection models, but their models were broader and more accurate because of all the new insight. In the end, this customer took a process that once took about three weeks from when a transaction hit the transaction switch until when it was actually available for their fraud teams to work on, and turned

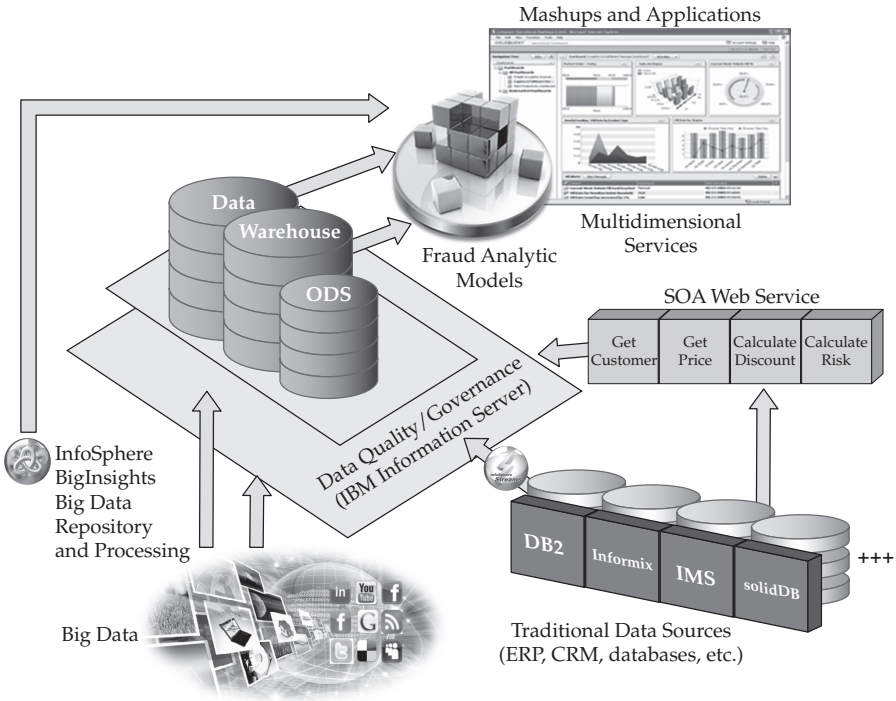

Figure 2-2 *A modern-day fraud detection ecosystem synergizes a Big Data platform with traditional processes.*

that latency into a couple of hours. In addition, the fraud detection models were built on an expanded amount of data that was roughly 50 percent broader than the previous set of data. As we can see in this example, all of that "80 percent of the data" that we talked about not being used wasn't all valuable in the end, but they found out what data had value and what didn't, in a cost-effective and efficient manner, using the BigInsights platform. Now, of course, once you have your fraud models built, you'll want to put them into action to try and prevent the fraud in the first place. Recovery rates for fraud are dismal in all industries, so it's best to prevent it versus discover it and try to recover the funds post-fraud. This is where InfoSphere Streams comes into play as you can see in Figure 2-2. Typically, fraud detection works after a transaction gets stored only to get pulled out of storage and analyzed; storing something to instantly pull it back out again feels like latency to us. With Streams, you can apply your fraud detection models as the transaction is happening.