OPERATIONALIZING ANALYTIC ASSETS

Building predictive models with deployment in mind

by Robin Way

CORIOS REDPAPER — Papers that get read.™

CORIOS **RED**PAPER — Papers that get read.™

OPERATIONALIZING ANALYTIC ASSETS

Building predictive models with deployment in mind

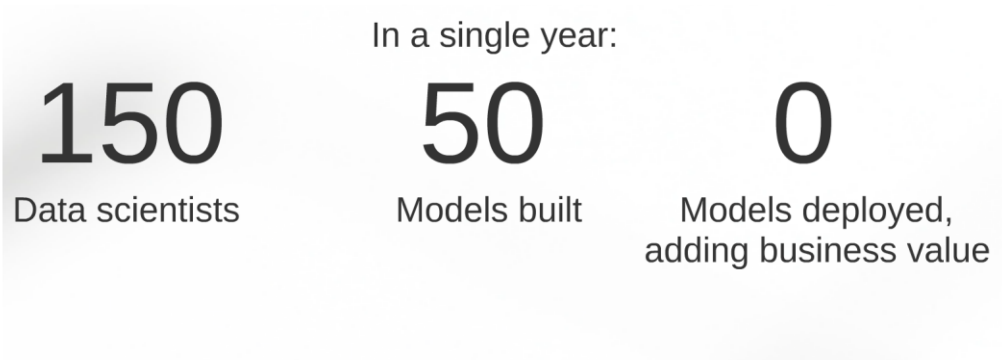
by Robin Way

Predictive assets aren't academic exercises

I've been a predictive modeler and econometrician for over 30 years, and the biggest challenge that confronts predictive modelers and data scientists is that it doesn't matter how good any model is, if it isn't used by other people in your organization to improve the way they run your business.

The current state of model deployment

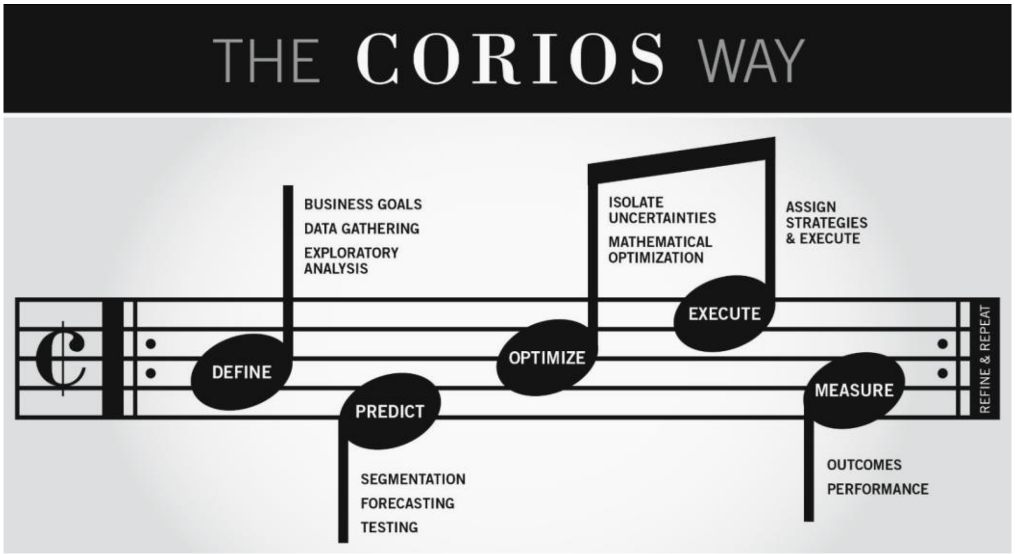
Most models don't meet this criterion. A recent Harvard Business Review article noted at a recent industry conference, that out of 150 data scientists, roughly a third had developed a model in the past year, but not a single one had deployed it into production, nor monitored that model's effect on business value. That's just sad.



Source: Harvard Business Review, 28 December 2016

We believe there are some common reasons that predictive models struggle to get used. They're complex, they're arcane, and there is no common process or vocabulary in most organizations that enables people across the domains of data science, customer-facing representatives, and technology to make daily business decisions using models. That's a problem we are committed to solving.

At Corios, our clients rely on making evidence-driven decisions using data and models, so it's vital that they build and maintain a healthy predictive model lifecycle, and continually upgrade their models as they lose efficacy. The state of the art is no longer focused on building the one very best model of its kind, and taking six months to do so, working at a craftsman's workbench, utilizing the latest in machine learning algorithms. It's entirely about building a repeatable, explainable, closed-loop process that creates a catalog, even a portfolio of models, each mapped to a set of treatments focused on predicting the behavior of each of a portfolio of customers. This is the Corios Way.



That closed-loop repetition is what we mean at the right side of this musical notation graph, with the coda that reads "refine and repeat". Your first model won't be the very best, nor your second, but over time, with enough experience, your portfolio of models will outperform the performance of your competitors who have stuck with the craftsman or technician approach and only built a single generation of a few models, while you have constructed a continuous release cycle of tens or hundreds of such models.

We'd like to help the industry define the discipline of rapid model development and deployment. Our own internal name for this discipline is Corios Tempo, which reveals that we think maintaining a stable rhythm of activity is an important design element. In this presentation, we'll discuss the problem we aim to solve, the design principles of a robust solution, the benefits of doing so, the roles and responsibilities inside the analytics organization that need to be harnessed to solve this challenge, the way we organize data, how we build models, how we validate and deploy them, how we administer the platform on which all this asset development and management activity takes place, and the best practices that we have developed over my 30-year career in analytics as well as the decades of collective experience developed by the Corios team.

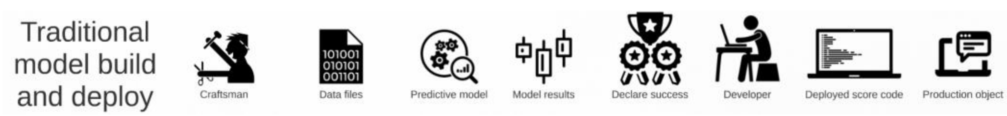
Traditional predictive model building

First let's address the nature of the problems to be solved. The traditional model build process is led by a craftsman, whose goal is to build a great model using tools of their own choice for a unique solution. They gather a variety of data files, build a predictive model, review its results, and declare victory.

What's wrong with this process? They often invest so much time in gathering their data, that they have very limited time to build a variety of models, each of which might have varying degrees of quality or effectiveness. Since they are operating in a vacuum, their ability to determine how good the model needs to be to perform its role is myopic; it might very well be up to the modeler to determine how good is good enough. And very often, the craftsman is dedicated to their craft, which is an admirable dedication, but what happens if the enterprise actually needs 10 models built, not just one? Or if an 80/20 solution would add substantial value compared to the status quo, but the craftsman wants to build a model that predicts the accurate result 99% of the time, which might either be impossible, or it will take 3 extra months to build, or it will ignore the inflated false positive rate that such efforts often incur.

Traditional model development and deployment

Yet another likely outcome is that the model craftsman doesn't actively consider what is required to deploy the new model in a production system, so that hundreds or thousands of customer-facing representatives from the company can use the decisions recommended by the model to directly influence customer relationships every single day, preferably using up to date information about the customer and their interaction with the company. and this is what drives the need to expand this process, by involving a technology-oriented developer to deploy model scoring logic, ensure the availability of production-level data files, perform the system integration and perform the validation and testing necessary to convert the raw model into a part of a production system that is widely available and performant. This new set of work is intensive, detail-oriented, and expands the skill sets required from just one person to far more: perhaps as little as 3 new people in the process of model deployment, and as many as 10. This means that the entire process is vastly more complex. It also takes much more calendar time, coordination and formal review.



Solution to this problem: the model factory

Is there a better way? Yes, absolutely there is. This is the model factory process. If the organization is going to need to deploy 10 or more models in a calendar year, or has at least 3 model developers, then they ought to consider building a model factory process. For some companies, this is a lot more activity than they currently experience; other companies build and deploy 10 models in as little as a week.

When using the model factory process, it starts with a set of model development and deployment engineers, rather than craftsmen. This is actually a more subtle shift than it might appear; the primary difference is in the set of expectations, in that the modeler should expect to be involved in creating a functional asset, not a piece of art, which needs attention paid to the trade-offs involved in design, development, testing, validation, deployment and maintenance.

The data that drives both model exploration, development, testing and production scoring and decisioning needs to be built to specifications that are more expansive than just model development. The expectation should be that the model engineer will individually build quite a few variations of each model before determining the difference between challengers and champion models, and the determination of model champions will be based on repeatable, standardized model selection criteria that are both quantitative and business-oriented in nature.

The model champions will be deployed into testing and change management processes so that it can be deployed in production.

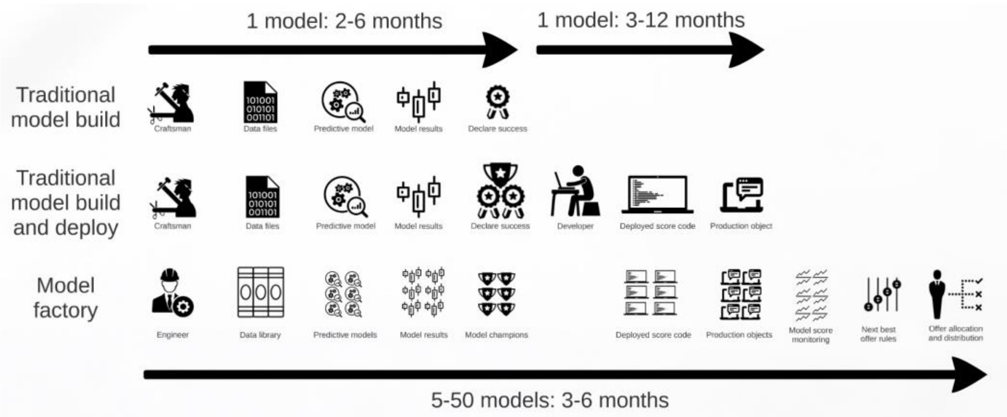
Unlike with the traditional process, where the craftsman and the developer are on different teams, using different skills, vocabularies, tools and platforms, the model factory engineer is involved throughout the process, to ensure that the complete development lifecycle for the model asset is in their purview. A more formal and standardized process of model performance monitoring, over time and across all model variations, supports ongoing maintenance for those models whose performance is out of acceptable range.

Furthermore, the usage of this large pool of champion models is often used in making next best offer decisions and in routing customer sales, service and pricing decisions on a routinely refreshed basis. This ensures that the champion models are collectively utilized in a next best offer process that ensures there is high business value attributed back to these model assets.

How do these processes compare?

What's the comparison between the traditional process and the model factory process in terms of throughput? Our experience indicates that the traditional craftsman process to build a single model can range from 2 to 6 months; deployment of a single model, depending on the care, planning and integration between the craftsman and the developer, is a lot longer and can range from 3 to 12 months per model.

Of course, most organizations recognize this length of time just isn't acceptable. What we have found is that the model factory process dramatically streamlines this process so that multiple models can be developed and deployed in as little as 3 to 6 months. The model factory takes more time to build, but once it's up and running, the changes in the company's ability to provide accurate and responsive strategies and treatments for its customers are revolutionary.



Model factory principles

The principles by which Corios designs, builds and teaches the model factory process include five key elements: closed loop, play nice, stop translating, single platform, and web services.

Closed loop

Closed loop means that the customer transactions, interactions and the presentment and response to customer offers all follow cycles. The effect of these interactions changes the nature of the offers we will present tomorrow. There needs to be a closed loop that streamlines the flow of customer behavior data through the model development and deployment process that never stops.

Play nice

Play nice means all the members of the model factory process need to be on the same team, share common goals, vocabularies and a commitment to not just throw assets over the wall to each other. We want to break down the silos between data providers, modeling analysts, systems integrators and change management owners.

Stop translating

Stop translating means not only that all the members of the model factory engineering team need to speak the same language with each other, but also that the model assets themselves use the same code base from model development through deployment, validation and integration with customer-facing systems and touchpoints.

One of the chief reasons that model deployment traditionally requires so much calendar time is that the model craftsman uses one code base while the developer uses another code base, and the systems integrator often uses yet another in turn. The developer literally translates the craftsman's code into a different language.

What happens next is, the developer asks the craftsman to validate the translation, which the craftsman can't do, because they don't write code in that language. And there is no re-usability of the translated deployable code; because the craftsman is going to build his next release using the tools he likes to use, regardless of what the developer expects. And the vicious cycle continues. This is also unacceptable. We need to stop translating. How can we accomplish that?

Single platform

Put everyone on the same, single platform for model development, deployment, validation and integration with customer touchpoints. Sounds easy, you say? Over the last 3 to 5 years, the model development discipline has become rapidly fragmented with the entry of all sorts of modeling languages, toolsets and platforms; many of them innovative, exciting and focused on some niche problems.

At Corios, we've become more and more convinced that this is the wrong direction to go, if your goal is to build a robust model factory. There are a very few platforms that can and have successfully delivered on all phases of the model factory, while giving everyone a common vocabulary and a way to play nicely with each other, and we've focused on the biggest and oldest one of them all, because it works every single time.

Web services for interoperability

Finally, adopting web services as a means to interoperate our model factory with the places where we interact with customers, means that we can have a lasting and widespread positive impact on customer relationships that also builds value for the company.

Benefits of the model factory strategy

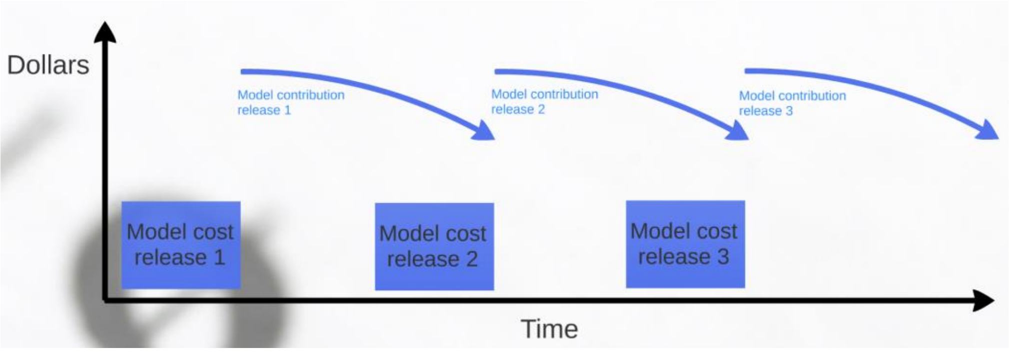
There are some substantial and practical monetary benefits to building the model factory that augment the impacts of faster throughput.

Decision-making cadence of the business

First we need to consider the decision-making cadence of the organization and its customers. There's no need to deploy and score customers faster than they interact with the company, or faster than any update in the score will actually change the recommended interaction or decision about the next step in the customer relationship. So whenever we talk about time in the context of the model factory, we're really talking about time relative to the customer interaction and decisioning cadence.

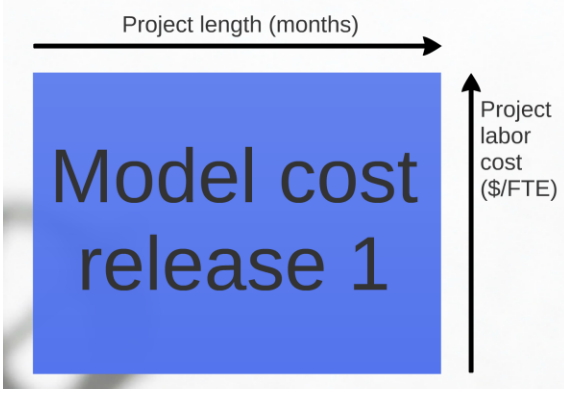
Why is faster better?

Second, faster is better because the company will make more money doing so. Here's how. When we build a release of a model and deploy it, it requires money (as measured by people's time and opportunity cost) for every tick of the clock invested into building the model release. Once the model has been put into production, it produces a revenue stream for the company. What we're looking for is for the model to produce a larger contribution to revenue than it cost to build.

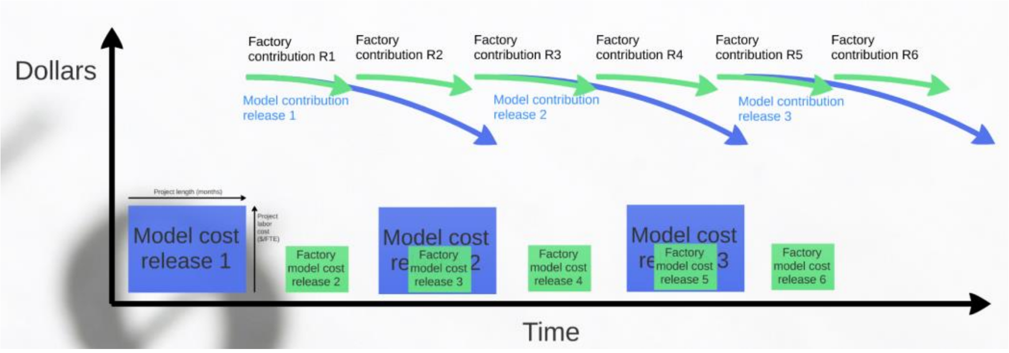


However, customer behavior migrates, and customer tastes and preferences change, so the model that we built today won't be as effective a year from now. The model ages, and needs ongoing maintenance, and eventually needs to be tuned up or even replaced. So we start a new model development cycle. However, we won't be ready to release a new model version for some time, so the existing model continues to age and lose its efficacy. Once the new release is finally ready, we'll deploy it, and start to enjoy that new car smell for a while, but the same cycle will kick in.

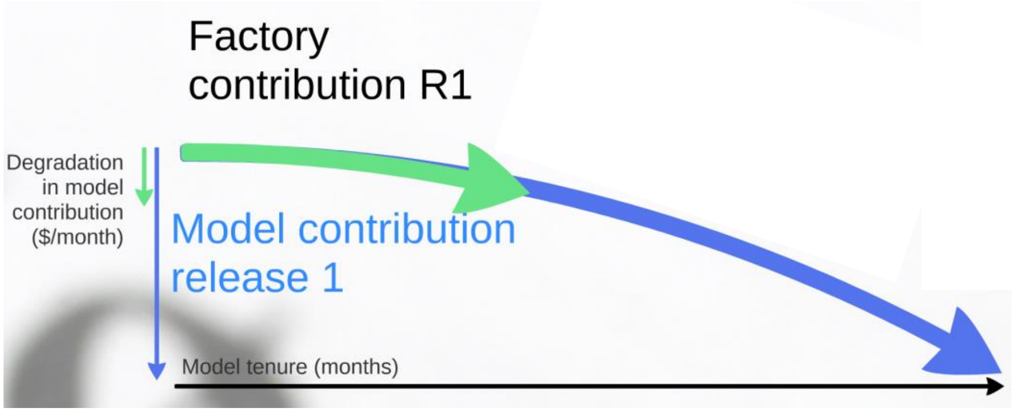
Less analytically mature organizations will continue this cycle of building and replacing models without getting more efficient at maintenance or more strategic about the investment required to manage this process. The cost of building each model is essentially the time it takes to build, multiplied by the labor cost per unit of time.



Smart, mature analytics organizations will invest in the model factory, where the very first instance of a new model might require the same time to develop the first release, (series of clicks) but each incremental release costs substantially less, and the cycle times are much faster. This means that the degradation in model contribution towards valuable decisions for each model release will be smaller, since the in-market time for an aging model is reduced.



So even though we might build and release more models in the same period of time, we save significant money on model build and release cycles, and we save an incredible amount of money by substantially reducing the degradation of value in that model's impact on our decisions.



The more models we have, the slower the release cycles in business as usual, and the larger the losses due to less effective decisions, the higher the return on investment for building a model factory.

Costs accrued by the traditional process

Let's evaluate how this accrues value for the company. Starting with the traditional approach, let's say that each traditional model build and deploy cycle takes 6 months, which itself is generous compared to most organizations' track record. At roughly $10,000 per month for each of 3 FTE to build and deploy that model, our cost for the first release cycle is $108,000. Across three release cycles, our build and deploy cost is $324,000, because each release is the same cost as the first.

Each model release will have a tenure of 12 months, for a total of 36 months, but with that long tenure, the degradation in model performance and value is an average $25,000 per month. That accounts for a $900,000 loss in model value across the 3 years. The total cost to the organization in the traditional approach is the build and deploy cost plus the model degradation cost or $1.2 million.

Traditional

Traditional	Time to build and deploy (months)	Build and deploy cost/month ($000)	Build and deploy cost ($000)	Model tenure (months)	Average degradation cost/month ($000)	Degradation cost ($000)	Total cost ($000)
Release 1	6	$ 18	$ 108	12	$ 25	$ 300	$ 408
Release 2	6	$ 18	$ 108	12	$ 25	$ 300	$ 408
Release 3	6	$ 18	$ 108	12	$ 25	$ 300	$ 408
Total cost			$ 324	36		$ 900	$ 1,224

Cost reductions provided by the model factory approach

Let's compare that with the model factory approach. Assuming that our first release costs the same as the traditional model, but we'll be able to shed an FTE and cut our release cycle length from 6 months to 2 months. Now our 36-month build and deploy costs are only $124,000, for a savings of $200,000.

Even more important, our model degradation costs will benefit from a shorter tenure per model, will lose only $5,000 per month instead of $25,000 on average per month, and our degradation costs are reduced ten-fold.

Our total build, deploy and degradation costs will have shrunk over that 36-month term from $1.2 million to a little over $200,000 for the model factory approach. And of course, this only represents a single model.

Model factory

Model factory	Time to build and deploy (months)	Build and deploy cost/month ($000)	Build and deploy cost ($000)	Model tenure (months)	Average degradation cost/month ($000)	Degradation cost ($000)	Total cost ($000)
Release 1	6	$ 18	$ 108	6	$ 5	$ 30	$ 138
Release 2	2	$ 4	$ 8	6	$ 5	$ 30	$ 38
Release 3	2	$ 4	$ 8	6	$ 5	$ 30	$ 38
Release 4	2	$ 4	$ 8	6	$ 5	$ 30	$ 38
Release 5	2	$ 4	$ 8	6	$ 5	$ 30	$ 38
Release 6	2	$ 4	$ 8	6	$ 5	$ 30	$ 38
Total cost			$ 124	36		$ 90	$ 214

Since the model factory approach allows us to deploy far more models with roughly the same labor, the benefits to the company are vast. Conservatively, only with this single model scenario, any prudent investor would have to take a close look at the benefits of the model factory approach.

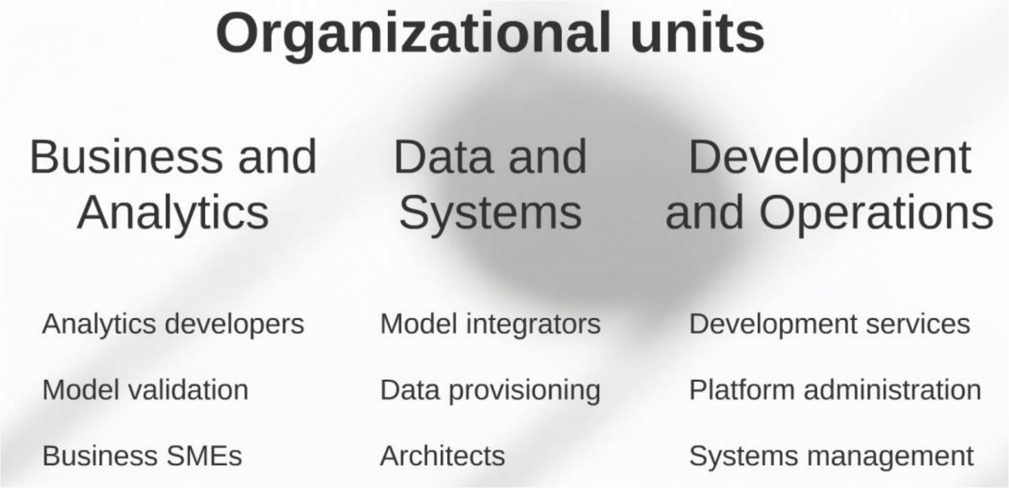
Building the model factory

Now let's turn our attention to the Corios approach that makes the model factory possible, starting with the team development model for roles and responsibilities.

Roles and responsibilities

We commonly work with three constituencies, other than the line of business who will be using the results of the models and next best action engines that rely on those models. These constituencies include Business and Analytics, Data and Systems, and Development and Operations.

The business and analytics teams are responsible for framing the problem and building the analytic asset; the data and systems teams are responsible for providing the infrastructure to perform the computations and to flow the business data through the model development and deployment process; the development and operations teams are responsible for migrating the analytic asset into production systems, supporting the interoperability of production systems that will use the scores and decisions provided by the analytic model asset, and owning responsibility for architectural design patterns that govern the entire analytic and technology process.

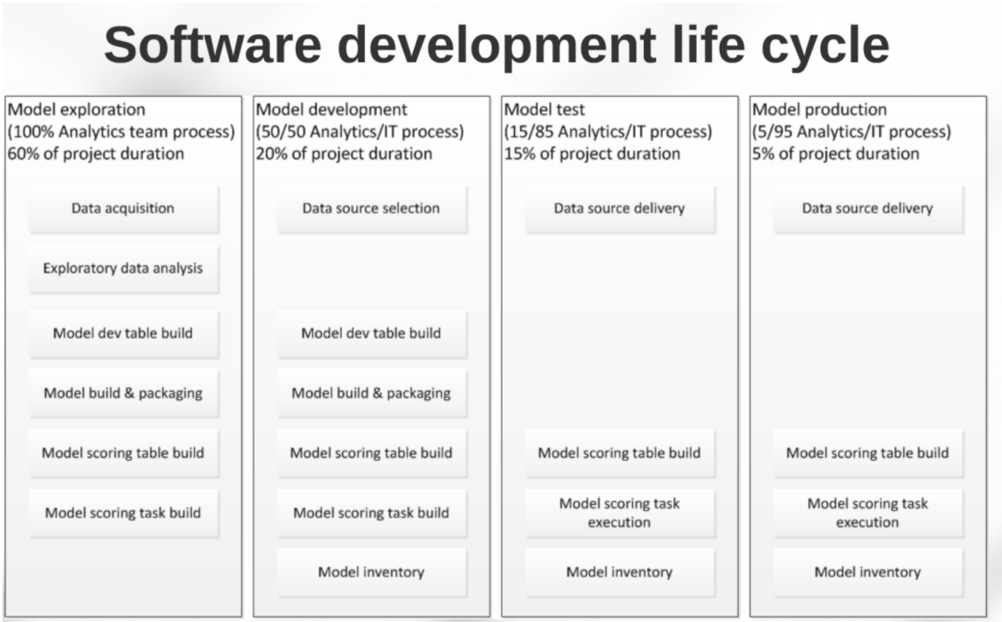


Analytic model deployment as a software development life cycle

It's really important to talk about all three of these constituencies as being part of one large team, but unfortunately this often isn't the case. We have found it's vital to create a common vocabulary and charter shared by all three of these teams. This is not a simple feat to achieve in most company cultures, where a bridge across these sub-cultures has rarely been established.

For this reason, Corios has adapted the traditional software development life cycle ("SDLC") to the concept of creating and deploying analytics assets, and most notably, we've added the Explore tier to the traditional tiers of Development, Test and Production.

The Explore tier of the SDLC is where the craftsman data scientist can still stretch their legs and let their mind grapple with the business problem to be solved. That said, we still believe it is vital to inject a repeatable process, so that even amongst the craftsman data scientists, there is a common process, language and conventions for exploratory data analytics that are shared and commonly referenced.



More detail on best practices in the exploratory data analytics ("EDA") process are available in the Corios EDA best practices paper (available on request). Furthermore, the soon-to-be-released Corios RedPaper on Analytics Change Management will be available from the Corios public web site by the end of 1Q17.

EXPLORE TIER

The analytics SDLC starts with the Explore tier, which is owned entirely by the Analytics and Business constituency, and is expected to require a substantial extent of calendar time, because model development is hard work. It's vital that the Explore tier is not treated like a Wild West adventure; the conventions and design patterns used in Development, Test and Production must be followed in Explore as well.

For instance, using a stray file of data downloaded from the internet for the purpose of Exploration, and then expecting that it can be seamlessly loaded in Production, is asking for big trouble. Even with syndicated data providers, whose file specifications evolve routinely, the version management of syndicated data files used in Explore should be identical to the specification used in Dev, Test and Prod. Corios has designed a series of practices and techniques for executing all these steps in the Explore tier of work, described below and also in supporting reference material.

DEVELOPMENT, TEST AND PRODUCTION TIERS

The next stages of the analytics SDLC introduce the gradual involvement of technology experts, including both the Data and Systems group and the Development and Operations group. This shared responsibility becomes more and more intensive on the technology teams as the model asset progresses through the SDLC. By the time the model asset reaches the Production stage, the business group is involved in a relatively small portion of the work. This is expected under the fabric of separation of duties.

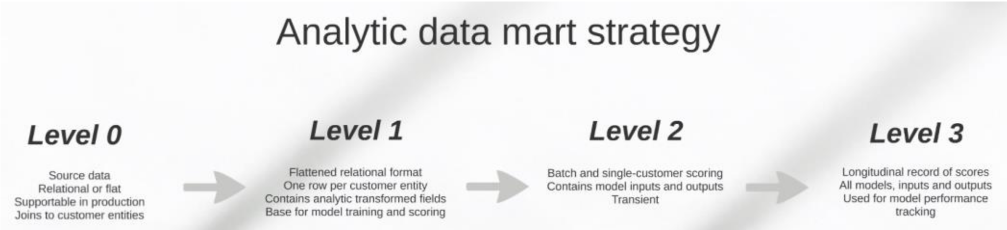
Corios has also designed technical practices that map to each stage of the SDLC workflow, as depicted in this schematic. Corios has embraced SAS as the primary technology platform for model development and deployment, and we have specified several sets of SAS tools for each stage of the workflow.

Corios GuardRails

In addition to sharing repeatable design patterns with all constituents, we've also identified anti-patterns, otherwise known as "what not to do". The patterns and anti-patterns associated with analytic asset development and deployment have been catalogued in a 50-page document that we call the Corios GuardRails. We've captured a large series of technical best practices for the use of specific analytics tools, secure coding practices applied to analytics asset development, and gathering and manipulating data files, as well as a formal charter of practices that are Required, Recommended, Optional, Disallowed and Under Review. We review these practices with members of all constituencies, especially the analytics and business users, because we often find that published standards on asset development simply aren't very common in most companies.

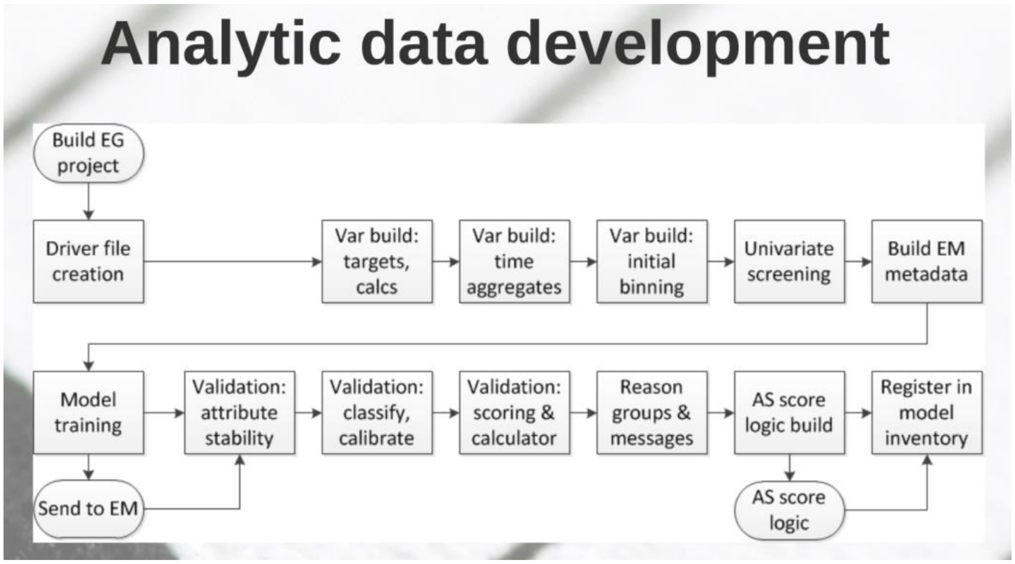
Analytic data modeling and storage strategy

The analytic data mart strategy comprises all information about customer entities from all relevant data sources. This strategy has four levels, which we've labeled Level 0 through Level 3. Each stage of data development plays a different purpose. Level 0 is used to commingle all common data extracts in one place, which can be supported in production. Level 0 is the foundation for assembling a coherent customer-matched file, the result of which is stored in Level 1. Level 1 is used for model development, scoring and initial historical validation. Level 2 is used for ongoing scoring, and contains both model inputs and outputs; this data level is transient because we're going to store its longitudinal results in Level 3. Level 3 is the place to capture all model scores for all models for all customer entities into the future, and is the source for model validation and model performance monitoring.

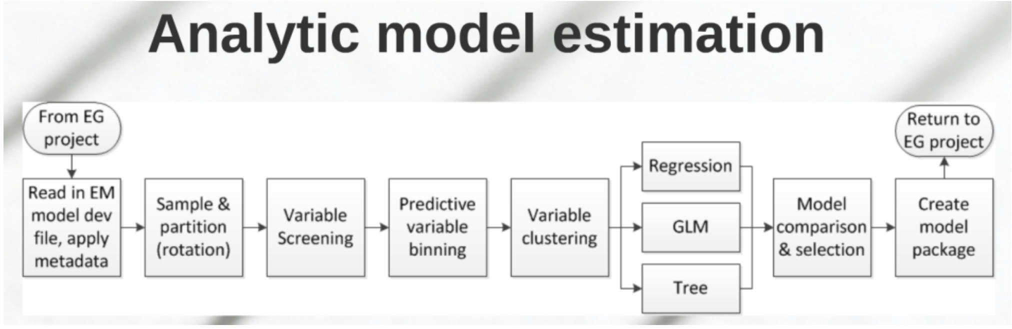


Model development processes

Model development processes include both analytic data development, which produces Level 1 data, and model estimation, which produces the model development assets to be used in producing Level 2 data as well as assets which we will store in the analytic model repository. This process is documented in further detail in the Corios best practices paper on Predictive Analytics Development.

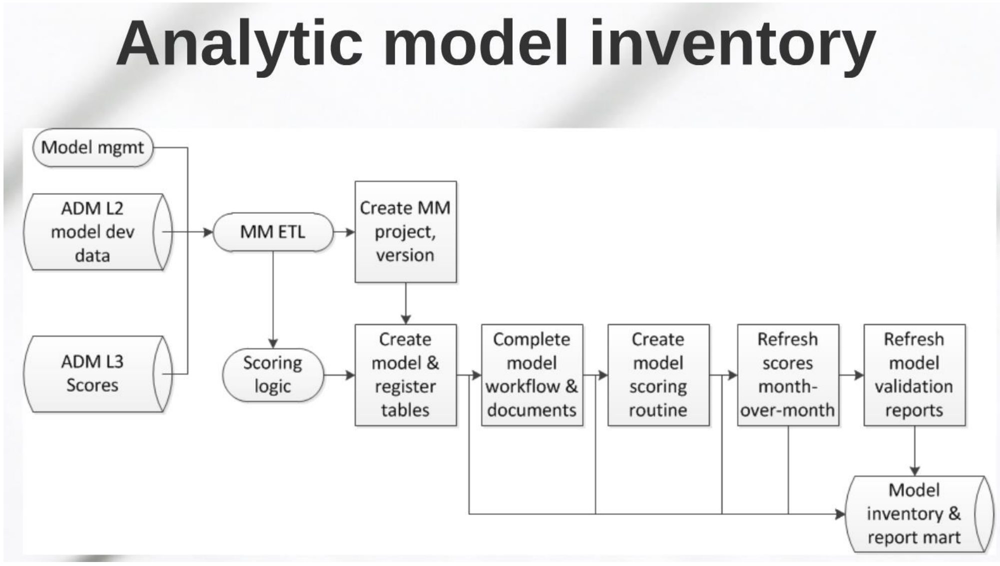


We've also developed a deeper dive on model development and deployment for banking credit risk models in a parallel best practices paper, which covers some of the fine points specific to this particular model development discipline.

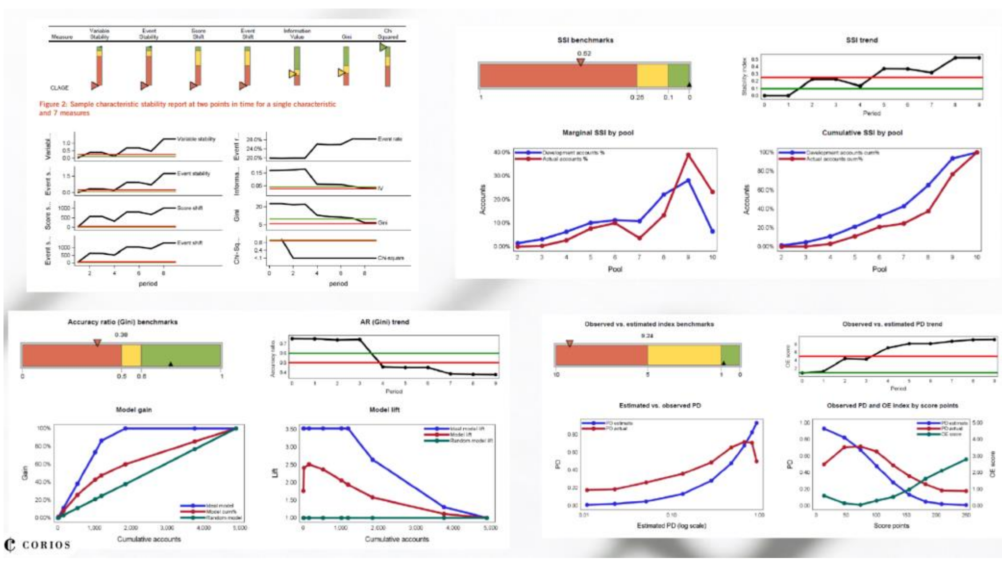


Model validation processes

Once the model asset has been created, it needs to be independently validated. Our process for model validation involves a specific workflow, process and technology stack. There are usually some very specific practices in place for model validation that include multiple constituents, review and approval workflows, documentation of findings and audit results, and quantitative and qualitative model attribute and performance reviews.



We've built a library of standardized model performance reports and dashboards, which we use on a wide range of model assets to assist with making model validation a faster and repeatable process. The standardization around the Level 3 data model makes the provisioning of data for model performance reporting dramatically faster and less painful.

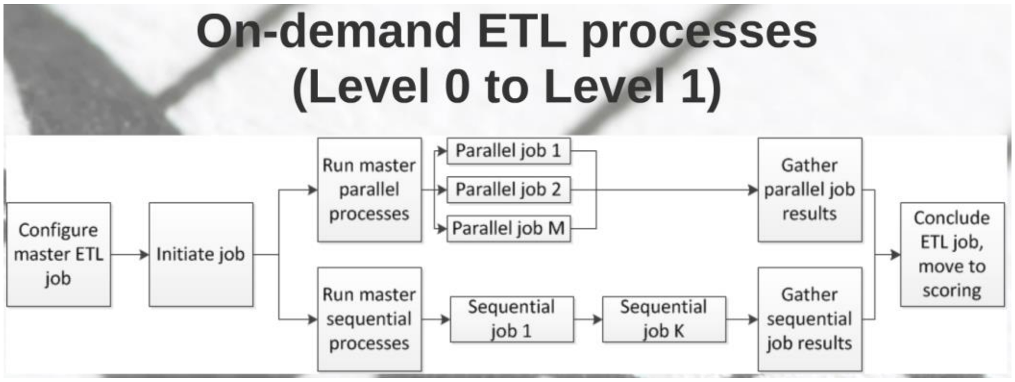


As with other phases of the model development and deployment process, Corios also has developed a substantial volume of best practices specific to model validation, as well as technology implementation practices, which are documented in the Corios RedPaper on Predictive Analytics Asset Management, available for download from our web site.

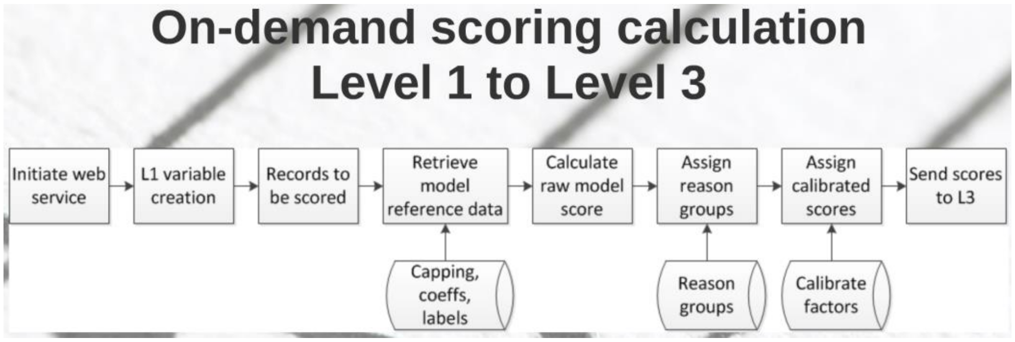
Model deployment and on-demand scoring

Once the model asset has been built and validated, it's time to deploy it into Test and then into Production. We typically find the most model assets need to be deployed so that they can score a single customer record in quote-end-quote real time, though we find that the terminology of "on-demand scoring" is both more descriptive and explanatory.

In the traditional model deployment experience of most companies who have tread ground here, this is where the science usually gives way to some mystical art, but we're doing our best to demystify this process. This process relies on the outputs of the Level 0 through Level 3 data development process, which is wonderful because we no longer need to build parallel data structures for our use in production that are different from the ones we use in Dev.



Some jobs in this process run in parallel, while others have to run sequentially, due to the nature of data transformations and their dependencies. Corios has developed a specialty in ETL design patterns that run effectively in an on-demand context, specifically utilizing the input/output design patterns of data appliances for asynchronous data access, and the use of web services for interoperability.



We also associate English business rule descriptions that provide context for the numeric model scores and decision data fields produced by model scoring jobs. A very technical but thorough demonstration of this on-demand scoring practice is available on our **web site**.

Corios written practices for model development and deployment

Corios is dedicated to sharing our best practices with clients in clear and transparent detail, both at the conceptual level and at a very technically deep and specific level as well. We've published many reports, best practice papers and Corios RedPapers that are associated with model development and deployment, and we're pleased to share them with our clients as well.

In summary, in this presentation we've summarized a number of the most important practices and conventions for successfully deploying predictive analytics in the enterprise. In everything that we do, we seek to guide the way for our clients towards more informed and evidence-driven outcomes that serve their customers' needs more effectively every day.

Conclusion

Want to learn more about Corios' capabilities for analytic model development and deployment and how they could be applied to your business? We'd love to share our experience with you as well. Email me at **president@coriosgroup.com** , or call me at 503-295-1685, and we'll get started today.

About the Author

Robin Way

Robin Way is the Lead Faculty Member for Banking at the International Institute of Analytics, and is the founder and CEO of the management analytics consultancy, Corios. He has over 25 years of experience in the design, development, execution, and improvement of applied analytics models for clients in the credit, payments, lending, brokerage, insurance and energy industries. Robin was previously employed with SAS® Institute's Financial Services Business Unit as a managing analytics consultant for 12 years, in addition to another 10+ years in analytic management roles for several client-side and consulting firms.

Robin's professional passion is devoted to democratizing and demystifying the science of applied analytics. His contributions to the field correspondingly emphasize statistical visualization, analytical data preparation, predictive modeling, time series forecasting, mathematical optimization applied to marketing, and risk management strategies. Robin's undergraduate degree from the University of California at Berkeley and his subsequent graduate-level coursework emphasized the analytical modeling of human and consumer behavior.

About Corios

Corios is a leader in the discipline of management analytics consulting focused on helping clients across industries to bridge the gap between their data and their business decisions. The company's custom offerings provide analytical solutions for risk management and compliance, marketing, pricing, and big data initiatives. Corios' solutions have identified business improvements worth hundreds of millions of dollars for corporations across North America.

Corios believes that data-driven techniques are the key to making profitable business decisions, and that data should be simple, approachable, and implementable for both analysts and decision makers. The Corios team is persistent in their efforts to promote transparency and common process across the analytical discipline.

To learn more about how Corios is bridging the gap between data and business decisions, please visit **coriosgroup.com**.

Operationalizing Analytic Assets:
Building Predictive Models
with Deployment in Mind

Published by Corios
133 SW Second Avenue
Suite 202
Portland, Oregon 97204

CORIOS

WE TELL THE STORY
NUMBERS CAN'T

Visit our website at: **coriosgroup.com**