RC25389 (WAT1306-042) June 17, 2013
Computer Science

IBM Research Report

Predicting the Likelihood of On-Time Delivery of Agile Projects Using ANDES

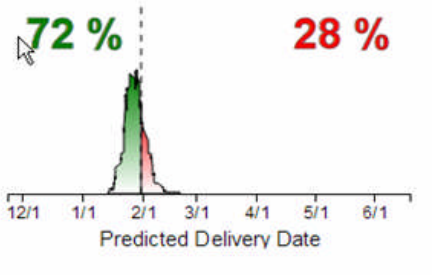
Evelyn Duesterwald
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Cambridge - Haifa - India - T. J. Watson - Tokyo - Zurich

Predicting the Likelihood of On-Time Delivery of Agile Projects Using ANDES

For questions/comments please contact Evelyn Duesterwald (duester@us.ibm.com)

Given the work required in a development project, specified as a set of tasks, ANDES predicts when the project is likely to deliver. As in many prediction problems, ANDES reasons about an *uncertain* future entity: the delivery date. The project delivery date is uncertain because it depends on a number of events whose occurrence we cannot know for sure, such as the completion of subtasks, the successful integration of components, etc. We usually can only take imprecise or incomplete measurements of such events. Thus, instead of modeling a single future delivery date, ANDES treats the delivery date as a *range* of dates, together with a probability function that provides the likelihood of delivering on each day in the range. Modeling the delivery date in this fashion, as a probability distribution, enables ANDES to reason about the *likelihood of delivery* by a certain date.



Consider the above example, which shows a probability distribution over a range of dates (approximately January 15 through February 20). At any date, the area under the probability distribution up to that date indicates the likelihood of delivering by that date. For example, the above curve shows a 72% likelihood that the project will deliver by February 1st, its planned delivery date. Conversely, there is a 28% chance that the project will be late.

Monte Carlo Simulation

At the core of ANDES is a work schedule simulation. This simulation takes as input a specification of the work as a set of tasks. There are numerous process-dependent variations on exactly what defines a "task". In ANDES, task refers to the smallest unit of work that a team explicitly breaks out and assigns to a developer. Tasks may have dependencies on other tasks, and they may have a number of task attributes, such as an owner, a priority/severity value, a due date, a type, and an effort estimate.

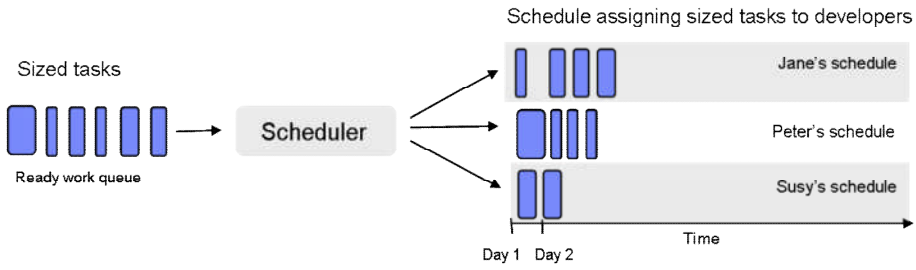
Many possible schedules can be generated for a given set of tasks, where a schedule is simply an ordered assignment of tasks to each available developer. The work schedule simulation proceeds by searching through the possible schedules. Each explored schedule obeys the constraints set by the task attributes, such as the dependencies, owner and due dates. The simulation randomly explores the space of possible schedules.

In order to explore a schedule, the simulation scheduler needs to know how long it will take to complete the task, that is, the *task completion effort*. Just as with overall project delivery date prediction, task completion effort is an uncertain entity. Hence, ANDES models task effort as a range of values together with a probability distribution, to which we will refer as *effort distribution*. For example, ANDES may estimate a given task as taking between 3 and 7 days, and it may be much more likely to take 6 days than 3 days. The width of an effort distribution expresses the degree of uncertainty in the estimate.

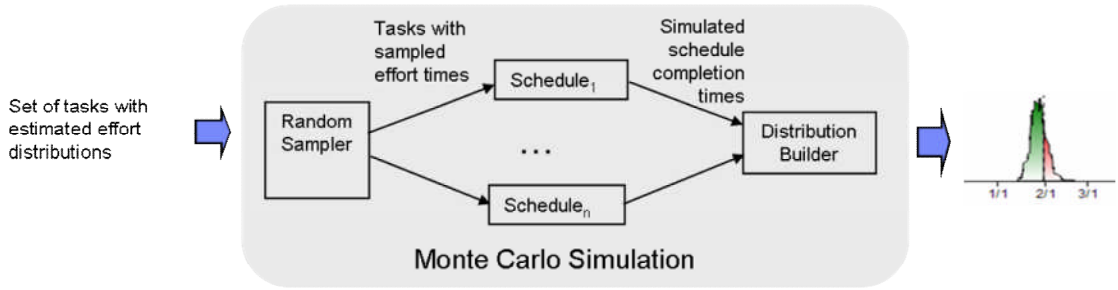
For example, for small highly predictable tasks, uncertainty should be low and the effort distribution will be narrow, covering a small range of possible completion times. Conversely, a large, complex task with a number of dependencies may have significant uncertainty. The corresponding effort distribution should be wide pointing to a large number of possible completion times.

We can now refine the input to the work schedule simulation as a set of tasks along with effort distributions for each task. Building a schedule out of tasks whose duration is specified by a distribution is most easily accomplished through the use of *Monte Carlo (MC) simulation*. MC simulation randomly explores a set of schedules in *n* simulation steps (say *n*=10,000).

During each simulation step, one schedule is built out of the available tasks. First, the available tasks are sized by randomly selecting an completion effort according to the task's effort distribution. The sized tasks can then be assigned using any existing job scheduling algorithm. We can use, for example, a simple list scheduling algorithm that maintains a ready queue of tasks (i.e., task that either have no dependents or whose dependents have already been assigned). If a ready task has an owner specified, it will be assigned to that owner. Otherwise, it will be assigned to a randomly chosen developer. After a task has been assigned, other tasks it depends on can be marked ready (unless they have other dependencies). The result of the simulation step is one schedule with one possible delivery date as shown below.



After *n* simulation steps, the MC simulation has built *n* schedules leading to *n* possible delivery times. Plotting these *n* delivery dates over time produces our delivery date distribution, as illustrated below:



The simulation does not try to identify optimal schedules, and it does not exhaustively explore the space of possible schedules. However, for a sufficiently large number of simulation steps (typically in the 10,000s) the simulation will have explored the most *likely* schedules leading to a determination of the *likely* range of delivery dates.

The accuracy and precision of the simulation outcome rests on the accuracy and precision of the input, that is, on accurate distributions of the level of effort. Thus, a main challenge in ANDES is the identification of realistic distributions of the level of effort required for task resolution.

A Machine Learner for Effort Prediction

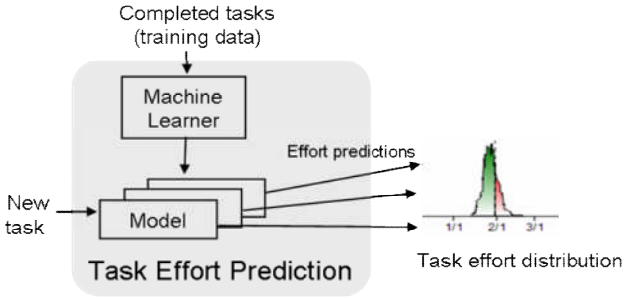
One way to obtain task effort distributions is simply to ask the developers to provide them. After all, developers may have a deep understanding of the complexities and risks involved in completing a task and thus may prove to be a good source of realistic effort distributions. However, in practice, developers often turn out to be poor and reluctant estimators. They may have personal estimation bias (e.g., be overly optimistic) or they may simply not have all relevant information available in order to make an educated guess. And when projects are in full swing many developers are unwilling to take the time to estimate their tasks. However, in the absence of any other source of information, for example at the beginning of a new development project, developer estimates might still be the most reliable source of effort distributions.

As the project proceeds, ANDES gains information about tasks and can begin to overcome the problems with user estimates using *machine learning* techniques. Machine learning can be deployed to predict task effort from the evidence that ANDES obtains from already-completed *similar* tasks. A key aspect of learning is determining what similar tasks are. The machine learner uses a training set of examples of completed tasks with their attributes including their actual completion times to build a prediction model. The prediction model discriminates the completed training tasks using a variety of task attributes (such as owner, type, or priority). Once the model is available, the machine learner can apply it to a new task to obtain a task effort prediction by matching the new task to the most similar training tasks.

For example, assume a scenario where tasks are either enhancements or defects. Consider a training set consisting of 10 completed tasks: 5 enhancement tasks that each took 2 days and 5 defect tasks that each took 1 day. From this training set of already-completed tasks, the machine learner might build a model that contextualizes its prediction depending on the type of task it is given. In this case, the model may simply encode that enhancements

usually take 2 days and defects 1 day. This model can now be applied to new tasks: if the new task is an enhancement, the model predicts 2 days, if it is a defect the prediction is 1 day. Clearly, this is an over-simplification: a real training set won't be as simple and bipolar, where different types of tasks always take exactly the same amount of time. To handle a more diverse (and realistic) training set, the machine learner may need to use a variety of attributes of the task (such as owner, task type, description, priority) in order to discriminate the elements in the training set to determine which ones are most similar to a new piece of work.

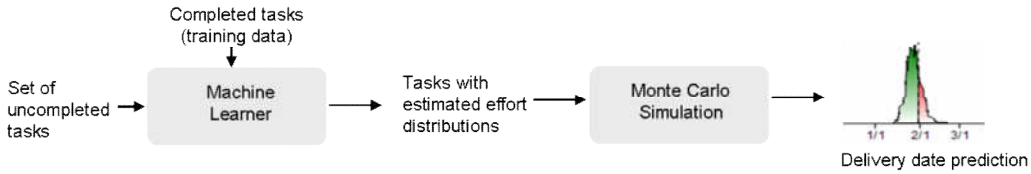
With many machine learning techniques, there is a tendency to *overfit*, which means that the technique will treat the training data as more representative of new data than it really is. To compensate for this tendency, the ANDES machine learner builds a series of models on different training sets, as illustrated below.



After an initial learning period, during which sufficient training data is gathered and the a series of models are built, ANDES begins to predict, for each as-yet incomplete task, an effort distribution.

Both team velocity (the amount of work a team completes in a given period of time) and the nature of tasks may change over time on a given project. Thus, it is important that machine learning is an ongoing process. Newly completed tasks serve as new training sets, and the machine learner continuously builds new models out of new training sets. As a result, task effort prediction is adaptive and reflects changes and trends that may occur during a project's evolution.

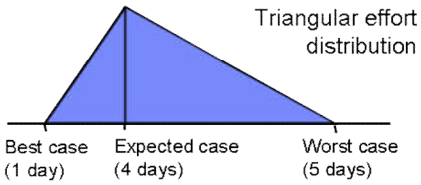
Combining the machine learning based task effort prediction with Monte Carlo simulation provides the following flow of the overall ANDES algorithm:



User Estimates

The machine learner, as described so far, does not require explicit user input; it learns solely from actual development data (the specification of completed tasks and their actual completion effort). However, there is an initial learning delay, during which training sets are collected and machine learning models are built. In order to provide prediction results during these early learning phases, ANDES can utilize user estimates of task effort, if they are available.

A convenient way for users to estimate task effort and to capture the uncertainty they may have regarding the level of effort is through the use of three values: a best case, an expected case and a worst case estimate. From these three values we can derive a *triangular distribution* of the effort as shown below:



If user estimates are unreliable, they will quickly be overcome by learned estimates as actual task completion time data becomes available for the machine learner. Conversely, if evidence suggests that user estimates are reliable, they will continue to form the basis of task effort prediction in the machine models.

ANDES obtains this effect automatically by feeding available user estimates to the machine learner as a task attribute to learn from. If it turns out that the estimates are a good indication of the actual task effort, the models will reflect this relationship and use the estimate as a primary discrimination factor for the prediction. If, on the other hand, the training set for the machine learner contains evidence suggesting that the estimates of a particular user are poor, they will be discarded and not utilized in the prediction models. This ability to adaptively shift between reliance on user estimates and reliance on machine predictions provides a level of robustness that can tolerate unreliable data sources, such as poor user estimates or unreliable development data quality.

Summary

- ANDES predicts the likelihood of on-time delivery as a probability distribution over possible delivery dates.
- ANDES uses a novel combination of machine learning and stochastic simulation techniques:
  - The Machine Learner trains on project history to build a work effort predictor
  - Monte Carlo Simulation is used to simulate likely project completion schedules
- ANDES is adaptive – ANDES continuously updates its predictions as new evidence becomes available.