



**A chief technology officer and committed environmentalist, Ryan Martens believes that agile practices and the software-as-a-service paradigm can contribute to conservation efforts by leading the IT industry away from a product-based, take-make-waste business model.**

As a long-time environmentalist, Ryan Martens has worked toward changing society from a take-make-waste model to a conservation model. In 2002, Martens, a civil engineer trained in project management, founded Rally Software Development, a venture-backed firm with 60 employees based in Boulder, Colo. As Rally's chief technology officer, he believes that Agile software development can help contribute to conservation efforts by turning information technology into a service-based proposition, rather than a product-based industry.

He spoke with *ProjectsAtWork* managing Editor Karen Klein recently; edited excerpts of the interview follow.

**What got you interested in the conservation movement?**

There was a lot of work that influenced me, starting in the late '90s and early 2000s. One idea was "[natural capitalism](#)," and particularly the book *The Natural Step for Business* [New Society Publishers, 1999] by Brian Nattress and Mary Altomare. A lot of people in the larger political world understand that there is going to be an end of oil at some point. And the transportation costs involved in using oil or coal to move things around this planet are way, way understated to the world.

**How does the conservation mindset play out in the world of software development and project management?**

Software development plays a role in the larger IT industry, and traditionally it's based around the product model, which follows a pattern of take-make-waste. We take raw materials from the environment or manufacture them; we make them into products; then we throw them away. This is not a good pattern for sustainability, but we got really good at it during the Industrial Revolution and we continually refined our ability to do it effectively. But because it's a resource efficiency and economic model that is not closed-loop, there's a negative impact on the environment.

**How does the closed-loop model in IT differ from what we have going on today?**

We're not using waste as a resource. Closing the loop means that you're not generating waste that's going to be discarded. So if you were designing for a full, closed-loop lifecycle, as a computer hardware manufacturer you'd take back your old products and figure out how to use them to make new products. You'd either melt the raw metal together and reuse it, or you'd find a way to reuse some of the computer chips or other components in the hardware industry.

**Are there instances of the closed-loop model being used in innovative ways in other industries?**

Sure. There are carpet manufacturers like Interface that lease their carpet, or sell it in squares. What you do is pop up the carpet tiles that are worn every few years, replace them with new ones, and take the old ones back to remake into new materials.

The same kinds of innovations can happen in IT. But right now we use material in producing hardware and software that is extremely toxic. Inside of computer monitors and silicon chips are thousands of toxic chemicals. Because product manufacturers sell these components to consumers, it's not their responsibility when you throw them away. So the cost of disposal isn't factored into the product price. It's like we don't care.

**How do you see that changing?**

I think we need to move from a product-based to a service-based economic model. By treating things as products, as opposed to services, we create a lot of dynamics. We transfer ownership of the product and the waste product, we treat costs as externalities and we're designing for miniaturization — as opposed to efficiency and environmental responsibility.

**Interesting. How did IT become a product-based industry in the first place?**

The only reason we're a product business is because when the PC business came about, we started shipping software products in computers. In the process, we shifted the burden and created a lot of waste – and we separated from the value of providing for our customers.

**How does Agile technology, and Rally Software specifically, try to change that model?**

Rally is a service company. We're a subscription business and we have to earn our keep each month. We don't actually send physical products to our customers, they don't install them, they don't have to patch them, they don't receive instruction manuals. Clients use our software over the Internet but it's used as a service. That way, we can deliver a continuous stream of value, make it easy for new customers to get started and continuing customers to use new features.

So we don't get a big up-front infusion of cash when a new customer buys our product. And our customer doesn't buy something physical that they're stuck with and that they must customize to meet their needs.

**What's the advantage of that — for the clients and for the environment?**

The product-based system is a terribly resource-inefficient model. In the software industry, that model operates on the concept of throwing something new over the wall before your competitors do it. And haste makes waste. But vendors, because of their large up-front costs and sales commissions, must produce software that derives revenue from upgrades. Otherwise, they have to continually produce new products in the space, which is difficult and very costly.

With a service-based model, we ship updates automatically every six weeks. We do minor things on a weekly basis that people don't even notice, like updating a defect. We also do conditional upgrades that allow the user to opt in if they want them. We are continually trying to bring in new subscribers and up-sell existing customers to purchase additional modules.

**What chance is there for the IT industry to move into an entirely service-based model?**

I don't see why the whole software service market won't flip by 2010. After all, salesforce.com has 700,000 users worldwide. Google, eBay and Yahoo all operate on this model. It's not technically impossible to do it. And there are the benefits of continuous value and leverage. If there's a problem or a defect with the software, the vendor is on the problem instantly — or else the customer's not going to stick around very long. The efficiencies of managing the defects from the vendor's side translate into huge positives for the vendor. I think with the ubiquity and bandwidth of the Internet and the various computing platforms out there, the software business can move the entire IT industry back to a service model.

**What about the hardware side?**

Hardware manufacturers like Apple, Dell and HP could close the loop on the entire IT lifecycle. Apple has already got some pretty interesting capabilities. Technically, I could sign up to be an Apple customer and they could ship me a new machine when I turned in my old one. And I'd get the new one as an exact replica of my old one, because everything on it is mirrored on the Internet. That kind of technology, if it's put into place, would allow the hardware manufacturers to change their business model, too. They could down-cycle their computers into schools and Third World countries, with some reconditioning. They could be in a completely different economic space than they are today.

**That's an optimistic big picture.**

Already there exist a lot of software service providers, or existing software firms making the transition. Agile really came alive as a process back in the mid-90s. It laid in the tools and the technology in the software space that get the costs of iterating down by breaking development into small, manageable chunks that can be released continuously. Since then there are a lot of things that have changed in the market, including how to scale up Agile so it works with large teams.

**How does the model affect project management?**

As we start building in Agile, project managers can start much more effectively looking for sources of waste and bottlenecks. Agile allows project managers to deliver working software in small, functional increments every two to

four weeks, which is ideally meshed with the service model. If we had to wait for a release vehicle for us to get those upgrades to customers, that would be disastrous, but in a product-based model, we would need that giant, big bang that would force customers to get their wallets out again.

**With increased awareness of global warming, there seems to be a determination in the popular imagination to do something about it. Do you see that playing into your efforts?**

I don't think that the worries about global warming are driving the software-to-service model. I do think that people are understanding that Internet-delivered models are very viable and much easier on the end user. But it's the business economics around software service that is starting to drive that.