



## White Paper

October, 2009

116 Research Drive, #122  
Bethlehem, PA 18015  
610-849-5056 (o)  
610-903-4274 (f)

## Knowledge as a Service Enables the Right Action, Right Now

*Success in any activity — from the practical to the creative — requires knowing what to do next. The key is having that knowledge at the time of maximum opportunity — the reason KaaS exists.*

**C**all it crowd sourcing meets software as a service — the Knowledge as a Service (KaaS) model brings together the two big ideas driving much of information technology’s recent innovation — to deliver on an even bigger idea. That is the ability to obtain the right knowledge whenever and wherever needed to perform a task successfully. Knowledge, of course, is the fuel that powers today’s knowledge-based economy — as opposed to information, which is what the average knowledge worker is drowning in.

The KaaS secret sauce is its ability to connect on-demand a specific information *need* (even if not fully expressed) with the correct information *response* — thereby producing knowledge in the head of the user. That takes *a lot* of information sources. Hence, the crowd, or cloud, which is to say a theoretically limitless number of subject matter experts dispersed throughout the Internet. And it also takes a delivery platform — equivalent to SaaS — or how dispersed servers respond to remote process requests. But instead of a software function, the cargo this time is information that becomes knowledge when: A) a user needs to know it; B) an expert knows what the user needs to know; and C) the two make a connection.

While KaaS could theoretically work anywhere knowledge works, early adoption is likely to occur in environments where knowledge is subject to rapid change and easily productized — such as IT system configuration and troubleshooting. In that environment, for example, knowledge is easily productized as helpdesk support, training videos, and tailored software — creating a natural marketplace with built-in incentives for providers and consumers to exchange knowledge for cash in real time. Such incentives are the gravitational attraction pulling players onto the platform absent the rigid and limiting formal structures of traditional knowledge engagement. (“Engagement” is a better word than “transfer” since hired experts often take their knowledge with them when they leave work.)

Since those limitations are absent, KaaS users see a number of significant benefits:

- **Anytime anywhere knowledge availability** — users can take action when they need to
- **Multiple experts can work on a problem at the same time** — answers are better and faster
- **Get only what you need** — resources aren’t wasted sifting through unneeded information
- **Pay only for what you need** — don’t waste money buying information you don’t need
- **Get expert knowledge, not just experts** — users get smarter and empowered

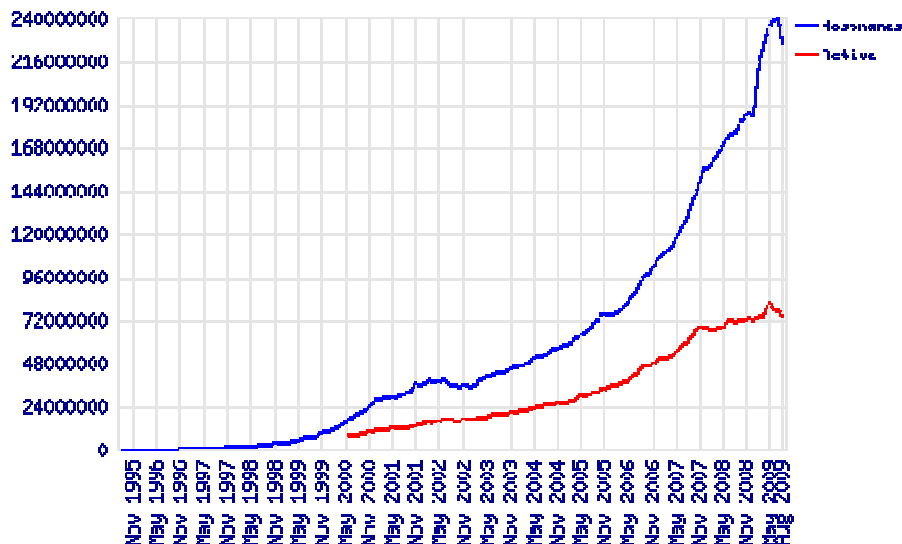
It takes more than money, however, to keep users or providers on any platform for long. Quality assurance will be key, both in terms of the product delivered and the experience of using the platform. Best practices are needed in areas such as ease of use and qualifying subject matter experts. Given those practices, however, KaaS is the answer for what users need to know now.

## A Compelling Business Case

Timing is everything in innovation. History is full of theoretically brilliant concepts that failed to catch on because people could not catch up. To resist change is human nature — so the bigger the innovation, the more compelling the business case required. The greatest impact occurs when a truly significant technology trend (or, better yet, several of them) coincides with an equally significant societal need. That is the case with KaaS.

In fact, when it comes to knowledge as a service, the pace of technology innovation and the logic of the business case feed off each other, almost in a closed feedback loop. That's because the need for knowledge accelerates at the rate that information grows — powered by the advance of technology, which is both a rapidly expanding *body* of information in its own right and also a rapidly expanding *channel* by which all information is delivered.

The information explosion and its effect on people's lives are self-evident. Consider just the growth in the number of websites — to almost 226 million in August 2009, according to Netcraft. In 2005, Google estimated that the total number of indexed web pages at over 8 billion — up from zero in 1990.<sup>1</sup>



And that is “just” the web. Overall, the total amount of knowledge in the world is doubling every 18 months, according to the American Society of Training and Documentation, while half of all we know today will be obsolete in 10 years.<sup>2</sup> The sheer volume and availability of information (because of technologies like the web) helps explain the massive growth in companies like

<sup>1</sup> Netcraft. (2009, September 23). *September 2009 Web Server Survey*. Retrieved October 1, 2009, from Netcraft.com: <http://www.netcraft.com>

<sup>2</sup> Geurts, T. *Supporting the Agile Enterprise in a Networked Knowledge Economy*. Utrecht: Be Value.

Google, which help people search information. If the amount of information were not overwhelming and not easily available, then services like Google would win far fewer customers.

And, by definition, the areas where information both becomes obsolete fastest and expands most rapidly are in fields of high innovation such as information technology. As IT continues to become ever more pervasive, more people need to know more about IT — which means they need to know a lot more a lot faster. They “need to” because IT exists to help people do things. As in any field, there are those who study IT simply for the personal enjoyment of knowing more about the subject. But for the vast majority, what they know about IT they know so they can perform a task — and that applies across the entire spectrum of modern living, from home entertainment to performing on the job in almost any profession you can name.

So, if you connect the dots, as in:

*Rapidly increasing innovation → (fuels rapid rise in information + rapidly growing need for information) → diminished capacity to act*

what you get is a self-sustaining chain reaction.

The blowback from rapid IT innovation is the need to further accelerate IT innovation — so people can power through this knowledge avalanche and actually *do* things. That’s where the “timely” and “the truly significant” technology trend comes in — or actually several of them.

## **Content Is What’s Relevant**

Today’s hot technology trends are hot for a reason. They are the locus of forces around Web 2.0: crowd sourcing, cloud computing, long tail, pull versus push, software as a service, virtualization ... among other trends. Like vectors on a map, these trends — in their variety, number and alignment — define the dimensions of individual self-enablement in a crowded information space. True, they are oriented toward different needs, but those different orientations allow them to triangulate on a common point. It’s all about how to apply highly dispersed network-based resources in highly scalable numbers in response to a particular individual’s request. Take a look:

*Crowd sourcing.* This is when a problem is presented via the web to a large number of potential solvers, any of whom will be paid if and only if they solve the problem. The solution is knowledge packaged in a form the user *learns* (like instructions or training) or does not learn (like software binaries). As a knowledge source, crowd sourcing’s strengths are the ability to scale the number of experts working on a problem and the ability to pay only for what you actually need.

*Cloud computing.* Where crowds refer to people, clouds in a Web 2.0 context refer to applications running on lots of servers spread across the Internet. The analogy to knowledge sharing is the economic benefit of having a resource (an expert or a software process) that you only pay for as needed, that is scalable, and which can be best of breed for addressing a specific need.

*Long tail.* Web-based bookstores like Amazon and Barnes & Noble make 90% of their money selling large numbers of titles — “the long tail” — that individually appeal to less than 10% of customers. The other 10% of revenue comes from best sellers. In other words, the vast majority of their customers are buying “information” only they and relatively few others want. The long tail concept translates easily to other kinds of information — like how to fix a computer problem. In the aggregate, the demand for such solutions is massive, even though only a handful of users might need any particular solution. By moving the information out of the physical bookstore, classroom or onsite consultant engagement, it is possible to serve the long tail and make money.

*Pull Versus Push.* Self-service is one of the reasons doing tasks on the web is so popular. People would rather just do it themselves than try to explain it to someone else. They have (or think they have) a better idea of what they want. So there is less waiting. And self-service is also usually cheaper. For the same reasons, many people would also just rather “pull” the information rather than have it “pushed” on them by someone else — Google, of course, being the obvious “pull” example. Users know (or think they know) what they don’t know. What’s different about knowledge self-service is that users are not always right. They don’t always know what they don’t know. Information alone doesn’t always enable them to act — if it’s the wrong information. That difference implies the need for expert mediation between user and information — someone to say, for example, “It sounds like to me that what you *really* need is ....”

*Software as a Service (SaaS).* Cloud computing resources are available from widely dispersed locations. One way to tap those resources is to bring them across the network and use them locally. Another is to use the resource remotely, but within a freestanding appliance. For example, if the resource is an accounting program, you can send the program your numbers, run the application and get back a result — pretty much sending your laundry out to be cleaned. A third way is to access the resource as a set of services — in other words, as pieces of the application’s functionality within a workflow that may also involve pieces (“services”) of other applications. That’s SaaS. But the same model works if the shared service is “only” knowledge. Whether the “backend” is human or software is an implementation detail — not a true conceptual difference. KaaS instead of SaaS still offers the economies of crowd sourcing long-tail content from the cloud in a pull-driven model.

*Virtualization.* This technology abstracts and decouples layers of computing (CPUs, operating systems, databases, applications, services, etc.) so it is possible to derive whatever combination best serves the needs of a particular user’s workflow. The advantage of a virtualized resource is the system’s ability to configure and deploy services to enable SaaS (or KaaS) functionality and economies. In a KaaS deployment, for example, the user doesn’t have to care whether there is a single expert or a dozen standing “behind the curtain” working on a problem — what software tools they are using, in what countries they happen to be located or whether one is an expert in databases, another in automotive supply chains and a third in data migration toolsets. From the user’s perspective, all that matters is the quality, timeliness and cost of the solution.

These trends don't just reflect the changing *how* of knowledge delivery — as in *methods*; they also reflect the changing *what* — as in *content*. For example, the distinction between knowledge that you learn and knowledge that you execute as binary code becomes less important. Take a term like “cloud sourcing,” which first applied to the sourcing of software code. It could just as easily replace “crowd sourcing” in any sentence when discussing getting solutions to users’ problems. This fading distinction is what Forrester Research analyst Holger Kisker talks about when he describes KaaS and Business Process-as-a-Service as both examples of “the innovation of business via collaboration ...” or what he calls the “...true potential of cloud computing.”<sup>3</sup> Business doesn't really care that software isn't human and therefore lacks self-awareness of the knowledge it contains.

What *does* become more important is the utility of knowledge in whatever form. Knowledge *is* knowledge — whether a conversation, video, book or piece of binary code — *because* it enables productive work. (Remove the word “productive” and this would be the definition of energy — a comment in itself on the evolutionary stage of today's economy.) The *how* is relevant to the extent that knowledge *quality* — as measured by its usefulness — is high. That is relevance to the person or organization doing the work.

## An SAP Knowledge Inflection Point

From how it works, to how it is configured, to various deployment scenarios — by definition there is a lot that can be learned about any knowledge-intensive product — and software modules are a clear example. But you don't need to know it all in order to apply the product successfully. A typical task's lifecycle involves multiple inflection points where the insertion of a particular “bite sized” piece of knowledge enables success until the next point in the cycle is reached, when another knowledge piece will need to be acquired, and so on.

A good example is applying SAP's Product Lifecycle Management (PLM) module while using cumulusIQ as a just-in-time knowledge resource. The scenario here illustrates five such knowledge inflection points, each involving a different department, throughout the lifecycle of what it takes to optimize route proposals. At each point, the knowledge required is different and so also can be the way that knowledge is presented for consumption. Success happens not just because all the possible information is available in the cloud, but also because the information is carved up in appropriate sized chunks and presented the right way.

1. **R&D:** “I want to learn how to become an expert at creating process elements in SAP PLM.

**Answer:** Find one-on-one interactive teaching in KaaS Trainer.

2. **System Design:** How do I distribute my SAP PLM recipe to a manufacturing plant?

**Answer:** Search KaaS Helpdesk FAQ for a short video or put the question to the cloud via Helpdesk Excellerator.

3. **Manufacturing:** I want to run a custom report on manufacturing performance.

**Answer:** Download custom code in KaaS Solutions.

4. **Quality Assurance:** “How do I configure the QA approval procedure?”

**Answer:** Search KaaS Helpdesk FAQ or put the question to the Helpdesk Excellerator.

5. **Shipping:** “How can I create the best route proposals for my specific business needs?”

**Answer:** Find consulting support in KaaS Trainer.

<sup>3</sup> Kisker, H. (2009, September 10). *The Rise of the Collaborative Cloud*. (F. Research, Producer) Retrieved October 1, 2009, from The Forrester Blog for Vendor Strategy Professionals: [http://blogs.forrester.com/vendor\\_strategy/2009/09/the-rise-of-the-collaborative-cloud.html](http://blogs.forrester.com/vendor_strategy/2009/09/the-rise-of-the-collaborative-cloud.html)

## “High Time for a Change”

If utility-of-content is the perennial *business* priority over container-of-delivery, it has only recently been a *spending* priority. Over two decades companies have spent billions on enterprise systems — for the software itself, for implementation services, for maintenance, and to reengineer business processes around these systems. It has been a “top down” push very much in opposition to the “bottom up” pull empowered by current trends. If someone needed some knowledge to do their job better — knowledge that system designers had not previously anticipated — that was too bad. Processes were reengineered for greater enterprise-wide consistency, greater reuse, less redundancy, a more efficient supply chain and a more focused alignment of internal activity on the external customer. And in that they have been hugely successful — at least to the extent their model of “the customer” was correct. But by enforcing consistency and alignment, they have also forced to the sidelines some of the conditions required for innovation in the face of that information avalanche left by prior innovation — conditions like individual self-enablement. Knowledge that users or customers need at any given moment cannot always be anticipated in advance — no matter how well constructed or uniform the containers. Ultimately it is what’s inside that counts.

This is more than just populist sentiment in the name of human potential. ERP companies’ huge Web 2.0 investments are not news. What *is* news lately is how these initiatives are bearing fruit with customers in the marketplace. Witness salesforce.com’s recent announcement that its Service Cloud offering has experienced 175% year-over-year revenue growth. According to the company, this is the “the first ever multi-tenant knowledge base designed for cloud computing” that allows users, including customer service reps, to access knowledge from cloud resources (including Twitter and Facebook) rather than simply articles in the corporation’s own database. Salesforce.com is not making this initiative to be trendy; but to achieve results like these:

Companies using the Service Cloud have seen a 28% increase in customer satisfaction, 25% increase in call deflection, 30% increase in first call resolution, 37% rise in service and support productivity and a 26% increase in customer retention, according to a recent third party survey.

States the company’s CEO Marc Benioff, “The customer service market is being held back by traditional technology. With two-thirds of customer service interactions moving to the cloud and the popularity of social networks, it is high time for a change.”<sup>4</sup>

---

<sup>4</sup> salesforce.com. (2009, September 9). *Salesforce.com Unveils the Next Chapter in the Customer Service Revolution with Service Cloud 2 - Leading Market Momentum and New Defining Technologies, including the Industry's First Knowledge-as-a-Service*. Retrieved September 20, 2009, from salesforce.com website: <http://www.salesforce.com/company/news-press/press-releases/2009/09/090909.jsp>

## Relevant Is What's Timely

If half of all knowledge will be obsolete in 10 years, the relevance of a particular piece of knowledge to a particular user is usually far, far briefer. The information may still be true; it may still be part of the long tail of *potentially* useful knowledge; it just may not be very useful for long. This is only common sense. Relevance happens when a user thinks something is relevant. That's the knowledge inflection point — the point at which knowledge most enables effective action.

### Key Differentiating Success Factor

SAP projects succeed for different reasons than they fail. That's according to three online polls taken in early 2009 — each of which garnered over 1000 responses. The poll questions and most popular responses were as follows:

**Poll 1:** What was the biggest reason that your ERP project wasn't as successful?

**Answer:** Organization buy-in/commitment

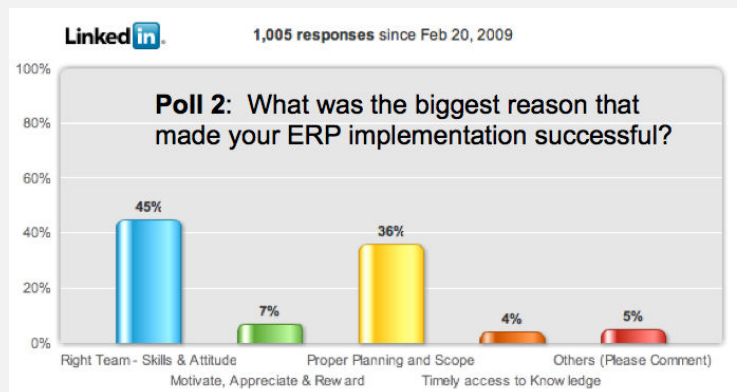
**Poll 2:** What was the biggest reason that made your ERP implementation successful?

**Answer:** Right Team — Skills & Attitude

**Poll 3:** What ERP projects are likely to succeed?

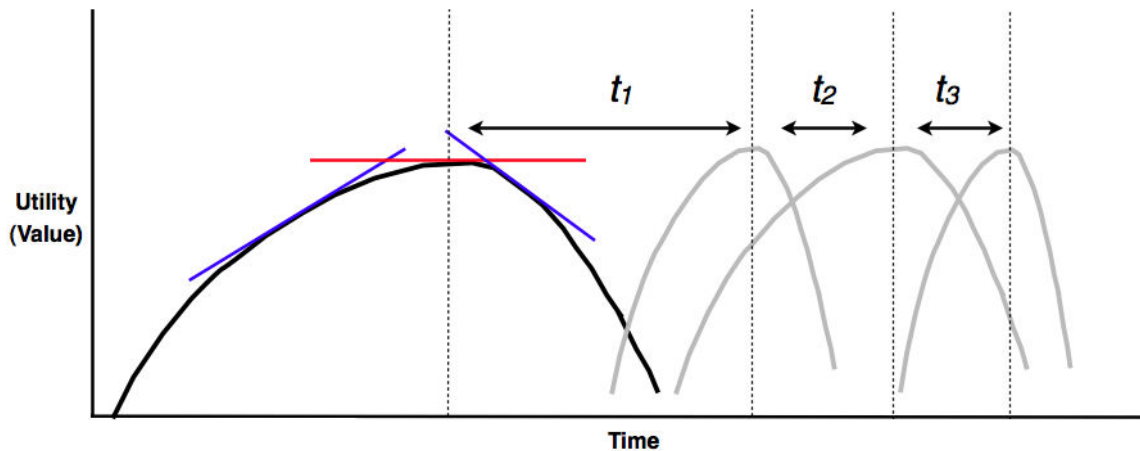
**Answer:** Knowledgeable in-house team

In other words, while some ingredients are *required* for success, they are not *sufficient* — a distinction easily overlooked. Of all the enabling factors, skills and knowledge make the most difference according to respondents. Furthermore, it is *in-house* knowledge, as opposed to knowledge possessed by outsiders (like consultants) that counts. That says that *having* relevant knowledge in-house when needed is the key differentiating success factor.



An inflection point — where the slope of a curve reverses direction — describes when the user starts gaining or losing value from a particular piece of knowledge. Depending on the situation, that change in value can go from negative to positive or positive to negative very fast. Acquired knowledge may also be retained for use in the future — unlike, say, a consultant who takes knowledge home at the end of a workday. So value is also retained. But the *relative* value declines (at the inflection point) as different knowledge becomes more relevant.

These points exist like gates along the path of a person's day. The right knowledge at that point makes it possible to proceed along the path — until reaching the next inflection point. This is particularly obvious in situations like IT configuration or operation. Typically a series of discrete steps are involved throughout the lifecycle of some task — all of which must be completed and in proper sequence for a task's success. The user does not need a complete course in the theory and operation of a given software system or how it could possibly apply in any number of vertical industries — and, indeed, all that information might get in the way. But what they do need to know, and know at a specific moment, is what to do *right now*.



The lifecycle of any process can be modeled as a series of knowledge inflection points for doing a series of tasks. These are points at which the slope of the curve representing the increasing utility of a piece of knowledge turns negative. The faster you can traverse the knowledge inflection points ( $t_1 \rightarrow t_2 \rightarrow t_3$ ), i.e., gain knowledge just in time to perform each task, the faster you can complete the process.

## Getting Relevant Content Now: and the Answer Is ...

So, what does it take to provide just-in-time relevant information (a.k.a. knowledge) to the user? As has been stated throughout, information regardless of form becomes knowledge when it is connected in the user's mind to a current need — the point of inflection. That connection requires the appropriate technologies, yes, and a cloud full of information. But it takes more. A reliable KaaS platform makes lots of these connections more or less automatically 24/7 for lots of users on demand and who may be working on their own. That presents several operational issues:

For example, users may be able to pull information on demand but they may not know what they don't know. Nor can they always filter good answers from not-so-good answers. So a KaaS platform requires quality controls — such as the ability to limit the pool of cloud-based subject matter experts to the most qualified. Obviously, you would also need to offer incentives to pull those experts onto the platform in the first place. Among the options: classic crowd sourcing models use prizes — set cash awards — for accepted answers.

Quality is also required in the area of user experience. For example, any platform that fails to deliver answers in a timely way obviously defeats the purpose for which it was intended. If information is to be useful, it must also be highly *usable*, such as having videos in popular widely supported formats. Another metric of usability is that information is packaged in whatever container best fits the user's need (rather than the reverse as is frequently the case on classic ERP systems, previously discussed). For some situations that might mean quick helpdesk answers, while for others it might mean a training video or a Java script.

Obviously, to pull the correct knowledge from the cloud, packaged in the most appropriate container, the user must be able to find it. If an expert doesn't make the connection for the user, such as in response to helpdesk questions, then where and how to look must be intuitively obvious — such as with robust search tools and clearly laid out subject matter categories.

The evolution of the KaaS platform is obviously a prime target of innovation in its own right. And with greater KaaS innovation will come ever faster, more appropriately targeted information to more users on a greater variety of subjects, and all with greater ease-of-use. What will be different this time is that this innovation will ease the burden of information overload, rather than simply add more to the problem.