HARVARD | BUSINESS | SCHOOL

9-610-032

REV: MAY 31, 2012

KARIM R. LAKHANI

DAVID A. GARVIN

ERIC LONSTEIN

TopCoder (A): Developing Software through Crowdsourcing

In December 2009, Jack Hughes, CEO and founder of TopCoder Inc., entered his company's headquarters in Glastonbury, Connecticut, eager to review a particularly complex software development project for an energy firm's dynamic power pricing system. Eight years after founding TopCoder, Hughes still enjoyed detailed project reviews. He was particularly proud that his company could produce high-quality software solutions for which his own employees did not have to write a single line of code. Instead, the firm nurtured a global community of more than 225,000 programmers who competed to design and create software modules for TopCoder clients, a process that the popular press called *crowdsourcing*.[1]

Hughes smiled at the project's success. The resulting software code was bug-free and operational on its first day, a rarity in the software industry. Especially impressive to Hughes was that in four months, 65 participants from 11 countries on six continents had competed in 57 contests to create this critical pricing system for the client (see **Exhibit 1**). As of 2009, TopCoder routinely produced software solutions for over 45 clients, including AOL, Best Buy, Eli Lilly, ESPN, GEICO, and the Royal Bank of Scotland.

In the past eight years, Hughes had refined TopCoder's business model to accommodate ongoing changes in the software industry, while also pursuing its unique competition-based software development approach. He had transitioned his business from a model that helped other software firms identify "top coders" to a company that developed custom software through a combination of traditional IT consulting services and competitions, mobilizing developers world-wide to solve clients' problems.

The shift to a greater emphasis on competitions, encompassing all aspects of software development, however, meant that project volume was a growing issue for TopCoder. Hughes had to think through how a competition-based business model, which increasingly stressed contests as an organizing as well as money-making approach, could handle increases in numbers of competitions, clients, and participants. Hughes considered his own goal: attaining $200 million in revenue from a high of just over $18 million in 2008. He fundamentally believed that contest demand would spur the supply of TopCoder participants, who would in turn create high-quality software solutions. But, was

[1] Jeff Howe, "The Rise of Crowdsourcing," *Wired Magazine* 14.06, June 2006.

Professors Karim R. Lakhani and David A. Garvin and Research Associate Eric Lonstein prepared this case. HBS cases are developed solely as the basis for class discussion. Cases are not intended to serve as endorsements, sources of primary data, or illustrations of effective or ineffective management.

$200 million in revenue a reasonable goal? Did his assumptions make sense? If so, what would it take to increase revenues by over an order of magnitude?

Background and Current Operations

Before he founded TopCoder in 2001, Hughes had built a custom software development[2] company, Business Data Services, in 1985; the company name changed to Tallan in 1991. Tallan employed some 600 people before being sold to CMGI in 2000.[3] As he was completing the transaction, Hughes reflected on what he had learned from his experiences at Tallan—the experiences that would inspire the core tenets of the TopCoder business model. Although Hughes enjoyed his time at Tallan, the company struggled in some areas. For example, recruitment was an expensive and frustrating process because finding qualified programmers was time-consuming and talent was difficult to assess. Due to constantly evolving technologies, programmers' skill sets often became obsolete after only a few years of productive service, leading to high levels of employee turnover. Furthermore, despite Tallan's goal of maximizing billable hours, Hughes believed there were opportunities to save clients time and money by, for example, reusing computer programs' basic components instead of building each application from scratch.

Drawing upon these and other insights, Hughes set about creating a new kind of organization that would build a "community" of programmers to help address the issues he had identified. These programmers would compete—as well as affiliate—by building and using components that had already been tested and found workable. The idea of reusing software components for new projects would become the core of the solutions the new company, called TopCoder, provided. Hughes envisioned the company as a "two-sided platform" for software development. One side of the platform would be clients, firms that needed software developed, who would work with his staff to specify programming challenges. The other side would be community members who would compete in contests to create solutions to the challenges for money and skill ratings. TopCoder would be in the middle as the platform host, designing and enforcing the rules of engagement between clients and the community members. Pete Bourdon, TopCoder's CFO, explained that the company needed to excel at five core tasks: breaking down large client software projects into components, taking in and processing client project specifications, determining appropriate contest prizes, having a consistent and unbiased way of selecting contest winners, and fixing bugs at the back end of development.

Setting out to amass an initial collection of highly skilled programmers, from 2001 to 2003 TopCoder asked established software development companies to sponsor world-wide web-based programming competitions. The sponsorships increased the popularity and legitimacy of TopCoder's competition platform and provided the company with access to talented programmers from around the world. In return, the sponsors, including Sun Microsystems and Google, used the contests to advertise and recruit new talent. Tanya Horgan, TopCoder's vice president of finance, explained that during the sponsorship phase, TopCoder offered unusually large prizes—as much as $5,000 to $10,000 per match for tournament winners—to attract competitors and expand the community. In addition, every contestant that participated received an objective numerical rating for their

[2] Custom software development by specialist firms in the global IT consulting and services sector (for example, Accenture and IBM) was an over $500 billion segment in 2008. (Source: "Global IT Consulting & Other Services: Industry Profile," *Data Monitor*, March 2009.)

[3] Clint Boulton, "CMGI Acquires Tallan for $920 Million," *InternetNews.com*, *February* 14, 2000, http://www.internetnews.com/ec-news/article.php/303771/CMGI-Acquires-Tallan-for-920-Million.htm.

performance against the global talent pool, providing a clear signal to TopCoder and others about the talent in the community.

By the end of 2004, the TopCoder community was 50,000 members strong. In its early efforts to use the community to generate revenue, TopCoder acted as a placement firm, matching top-rated community members with firms seeking new talent. Hughes, however, was dissatisfied: "I cringed at the idea of TopCoder becoming a placement firm. That was not my end vision for the company."

In 2005, TopCoder began to use its community to develop software components and applications. Hughes first tested this model by having highly rated community members compete to redesign and rewrite the firm's own software platform. The resulting code was higher quality and much less expensive than TopCoder's own internally developed solution. Hughes now had positive proof that complex software systems could be built through competitions.

Initially, TopCoder adopted a model to create solutions for clients by contracting with community members, running competitions, and providing consulting services. The company broke down the software development process into seven distinct but interrelated tasks: 1) conceptualization, 2) specification, 3) architecture, 4) component production, 5) application assembly, 6) certification, and 7) deployment. Most revenue came from consulting services: TopCoder billed clients for the time the company's platform managers spent conceptualizing and specifying client problems, setting up component design and development competitions, assembling components, and delivering finished solutions.

Shortly after TopCoder started developing software for clients, the company identified reusable components from the software it was creating and collected the components in a catalog. These software components became an important part of TopCoder's value proposition to its clients. Many of the custom applications could be produced by combining existing catalog components with new components built through competition. TopCoder had also received eight U.S. patents for various aspects of running online programming contests in a distributed community setting and had other patents pending domestically and internationally.

TopCoder's hybrid consulting model led to large increases in revenues. However, Hughes was still dissatisfied: "I viewed the hours-based services approach to be a broken, inefficient model." In 2007 and 2008, TopCoder produced nearly $20 million in revenue, but platform manager costs remained high (see **Exhibit 2** for information on revenue and platform manager costs). Attempting to alleviate costs, in 2007 TopCoder introduced competition tracks for component architecture and assembly. With these new competition tracks in place, the work traditionally done by platform managers would now be done by the community. In 2008, the company also added competitions in software development tasks, such as conceptualization and specification, as well as deployment and bug fixing.

By early 2009, TopCoder had moved increasingly away from the hybrid consulting model. It now focused on completing all tasks in software development through competitions. Instead of paying for time and materials for TopCoder platform managers, clients paid a monthly platform fee based on the complexity of their software requirements and the estimated number of competitions they would run through the TopCoder platform each month. The platform fee also provided clients with unlimited access to the over 1400 components in TopCoder's catalog. Roughly 60% of most clients' projects could be accomplished through reusing components from the catalog. The company coupled the move from the hybrid consultancy model to a competition model with the reduction of many platform manager positions, leaving the company with 16 project managers servicing 35 clients by the end of 2009.

As of late 2009, TopCoder ran two different types of competitions on its platform: algorithm and client software development. Algorithm competitions served as the primary means for attracting new members and retaining existing members. These competitions required members to develop creative software solutions to relatively difficult programming challenges. All members were assessed against each other through an automated computer scoring system; they then received a TopCoder rating for their performance. Some algorithm competitions also had cash prizes for winners.

The second type of competition targeted developing software applications for specific client needs. A TopCoder platform manager initially worked with the client staff to develop a "game plan" (see **Exhibit 3** for a representative game plan) or a project road map for building the software. The first step typically involved a contest where the general client problem was presented to the TopCoder community in a conceptualization contest. Here contestants publicly cross-examined the client staff as to their actual needs and then submitted a business requirements document and high-level use cases. The client chose the submission or submissions that best represented the client's needs as the basis for further development. Then a series of specification contests was held to create the application's requirements documents, application wireframes (i.e. the logical flow of the application), and storyboards (detailed cases of the user experience). Next, the output of the specification contests was fed into several architecture contests to create the overall system and component level designs. At this point, the TopCoder platform manager would work with the client to either select components from the catalog or commission the creation of new components through design and development competitions. After the component production phase, all the relevant components were put together through an assembly competition with the objective of creating a working system. Assembly was then followed by certification and testing contests and then, eventually, deployment. Throughout the execution of the game plan, TopCoder retained flexibility in development by running "bug races" to accommodate changing client specifications or unforeseen errors.

To determine winners and assess quality in client software development, TopCoder used a community-based peer-review system. In particular, expert and experienced TopCoder community members were paid to grade and comment on all contest submissions using detailed scorecards, ultimately picking the contest winners. The winning competitors for each contest then received monetary prizes, and all participants received updated ratings for their performance. TopCoder also ran studio contests if an application required logos or graphics; in those cases, clients chose the winners.

Evolution of the TopCoder Community

Growth and Composition

From 2001 to 2009, TopCoder added an average of 25,000 new computer programmers to its community each year. After filling out a short online registration form, anybody in the world could participate in a software development competition; by spring 2009, the TopCoder community had over 200,000 members (see **Exhibit 4** for community growth). Although the size of the overall community was large, the number of people within that community who actively participated in contests and posted in forums was much smaller. The majority of community members at TopCoder registered as members of the community but never competed in any contests. In fact, by 2009, only 35,000 unique individuals had competed in contests. To Mike Lydon, TopCoder's chief technology officer, the remaining 82.5% of the community was the "latent pool": people who were interested

enough in the TopCoder platform to register and had the potential to provide TopCoder with increased development under the right conditions.

A second group within the TopCoder community comprised those members who at one time participated in TopCoder contests but then stopped participating. Lydon noted that, after TopCoder decreased prize values in 2008, many competitors from the United States and Canada left the TopCoder community. Yet another group included people who participated in TopCoder contests but did not win. TopCoder saw those competitors as the "long tail"—people who primarily competed for the sake of learning. One of TopCoder's main goals was to cultivate the long tail so that less-skilled competitors could improve over time and increase their levels of contribution. Lastly, TopCoder's most valuable group of competitors included the everyday winners. The talent of TopCoder's elite programmers was equal to the best in the world, but such members only accounted for 0.5% of the total TopCoder population.

The core of TopCoder's community was made up of single, highly competitive males in their 20s. According to Michael Paweska, a six-year veteran at TopCoder: "To be successful at TopCoder, you must ask yourself, 'Are you a competitor?' You need to be able to thrive on competition; you can't be scared of it. You also need the flexibility to work long hours. TopCoder is a bachelor's sport: the moment you become involved with someone else, it becomes a point of friction." TopCoder attracted competitors from developed nations such as the United States, Canada, South Korea, and Japan, as well as from emerging economies such as China, Russia, Poland, India, and Ukraine. Wu Yanbo, a Chinese TopCoder community member studying abroad in Australia, explained that most competitors in the lower-paid contests were from developing countries. According to Wu, the prizes were not large enough for many individuals from developed countries to compete, since they could spend their time better elsewhere.

Justin Gasper, a member since 2001, began experimenting with the TopCoder platform while working for a traditional software engineering company. After winning significant money with TopCoder, Gasper decided to quit his job in 2005 and devote 40 to 50 hours a week to TopCoder. Gasper explained: "TopCoder is my full-time job; I don't have a day job." Gasper was one of TopCoder's regular winners, a member of the "global elite" of programmers. In architecture competitions, Gasper won at least second place 95% of the time and had a win percentage of 69.23%. Competitors at TopCoder could choose which contests and what type of contests to join (see **Exhibit 5** for participation and prize data by contest type).

Profiles and Ratings

Each programmer in the TopCoder community maintained a public profile that displayed his or her user name, contest history, and basic personal information. Another part of the member profile displayed a competitor's numeric rating for each type of contest. The rating system was modeled on the one used to rank grandmaster chess players engaged in worldwide competition. A "red color rating," or a rating of over 2,200, represented elite status within the community and a high skill level. Yellow, blue, and green color ratings represented descending skill levels. Each competitor's country rank, total community rank, success rates for contests, and *reliability*—or percentage of times the contestant joined a competition and submitted a passing solution—were featured in their profiles. TopCoder members could also choose whether or not to display their total earnings on their profiles (see **Exhibit 6** for an example member profile).

Motivating Members

Between 2001 and 2009, TopCoder paid out over $20 million in prizes and peer review money to its community of developers. However, prize money was not evenly distributed throughout the TopCoder community. The top 5% of prize earners received approximately 80% of the total prize pool, while the majority of TopCoder community members earned little or no money from competitions. Some competitors were extremely successful. For example, from 2006 to 2008, Paweska earned $200,000 to $300,000 per year, while Gasper averaged over $100,000 annually. Wu commented: "I have to say money is the most attractive thing. The prize is very good compared to the income of my friends who are working in some local companies in China. Even though the economy is not very good and TopCoder reduced its prizes, I can still earn around $1000 per month in my spare time." TopCoder typically awarded prizes to the top two submissions in each contest, with the lion's share of the prize money going to the top performer.

Besides prizes awarded on a contest-by-contest basis, another main source of income for members was the Digital Run. In the Digital Run system, the top five ranked competitors for each contest were awarded points based on contest rank and performance. At the end of each month, TopCoder tallied competitors' total points and awarded the top point earners thousands of dollars in bonus prizes. Paweska explained that success in the Digital Run was not all about who was the best programmer but more about who could handle the most all-nighters. Other competitors, such as Gasper, also made money through contracted projects that TopCoder assigned.

In addition to their cash earnings, many community members reported that their TopCoder rating was very important because it provided an objective assessment of ability. Wu commented that it was not easy to maintain a very high rating as it required familiarity with many kinds of technologies, quick thinking, the ability to learn independently, a strong work ethic, and attention to detail. According to Wu, a TopCoder rating could be important for a programmer's future career. For example, a high TopCoder rating helped one of Wu's friends earn a job at Google. Gasper noted that TopCoder ratings were also symbols of status and prestige for many programmers: "If you have red ratings, people look up to you." Indeed, many prestigious software firms asked potential recruits to get a TopCoder rating before applying for a job. To others, however, the rating system was less important. Gasper, for example, explained that winning and making money meant more to him than ratings.

Although there were differences of opinion regarding the importance of ratings, almost all community members agreed that competing at TopCoder provided numerous opportunities to learn and improve. In fact, for many programmers, a TopCoder career often began with failure, but post-contest evaluation and peer review of each submission helped them grow and improve. Gasper noted: "I totally failed in my first competition. But the reviewers were really good at pointing me in the right direction, saying 'here's where you went wrong' … You can't fake it because you're getting peer reviewed by people who are better at programming than you are. The reviewers don't care if they hurt your feelings; they are direct. If they see a bad design, they rip it apart." Paweska agreed that getting feedback from reviewers was crucial and added that community members could also learn from acting as a reviewer for contests. For scientists and developers, Wu believed that algorithm contests were particularly helpful at sharpening research skills and improving critical thinking abilities. In all cases, continual learning opportunities from peers were an important reason for participation.

Gasper described the appeal of working at home on a web-based platform instead of in a traditional "cubicle farm" setting: "I like the flexibility that TopCoder gives me. I don't need to drive

a half an hour to work each day and can do the same work at home. If I want to take off a day to play golf, I just do it. I also don't have to work from 2:00 to 6:00 p.m., my most unproductive hours." Sharing similar sentiments, Paweska liked that while working at TopCoder he did not have a supervisor looking over his shoulder.

Setting one's hours was convenient but also challenging, as competitors had to actively manage their individual levels of participation. Gasper constantly balanced effort and reward to maximize income while still living a sustainable lifestyle. "If there's something that is way too much work for the payment, I won't do it . . . that's a super power I've developed. I know when the spec is clean and worthwhile to solve. It's a skill that comes from doing tons and tons of contests."

A "Community" of Competitors

Wu noted that, although the firm was competitive in sprit, competition at TopCoder was never disrespectful or nasty and that people liked to help each other, even when they competed in the same arena. TopCoder forums were the main source for collaboration. In the forums, less-experienced community members asked for assistance on certain problems and received instant feedback from more-experienced competitors.

At TopCoder, conversations and relationships extended beyond the scope of software development. Hughes reflected on a particularly remarkable exhibit of communal strength and caring for fellow community members outside of software development. "When one of the community members died," he said, "the outpouring of support was such that a number of the community members took all of their winnings for a few weeks and gave it to the deceased's wife. It ended up being tens of thousands of dollars."

Once a year, TopCoder paid for all of the best talent from the community to travel to Las Vegas, Nevada, to compete in the TopCoder Open (TCO). In addition to serving as a proving ground for the best programmers in the world, the TCO provided community members with the opportunity to network professionally and socially.

The TopCoder community had a distinctive culture, with identifiable personalities. Wu explained: "I believe this community, like all others, has its own culture. Clearly, the members built it up continuously. When I joined the community, there were already some leading members who were active in competitions and forums, brought out good suggestions, and started up interesting and important discussions." In some cases, the fame of community leaders extended well beyond TopCoder. For example, Tomasz Czajka, from Poland, achieved "rock star" status and had his picture plastered on billboards throughout Warsaw after he won the TopCoder Open in 2006.

The Client's Perspective

Clients came to TopCoder to have high-quality software developed in a cost-effective and time-efficient manner. TopCoder positioned itself to serve both large firms and medium- to small-sized business that wanted to see systems developed. Keith Moore, a TopCoder client and former senior vice president at LendingTree.com, believed that, regardless of size, any company could take advantage of TopCoder, whether it was a five-person operation or large outsourcing vendor. For many CIOs, the process of software development and talent recruitment was a major headache, and missed deadlines and large cost overruns were common worries. According to Stephen Laster, the CIO at Harvard Business School and a TopCoder client, "A typical IT shop will turn over 48% of its employees every three years. This process is very costly. The same problem exists with our

outsourcing consultants. When selecting consultant teams, we tried out 60 programmers before finding our team of 20. With TopCoder, I pay for performance and the CIO sees Nirvana."

As of 2009, TopCoder had developed a strong relationship with existing clients for delivering high-quality software solutions and superior customer service. After completing their first project with TopCoder, 82% of clients signed up for a second round of contests. These clients cited several advantages.

Benefits

Better Ideas Before sinking thousands of dollars into a project, a client could run a conceptualization contest through which TopCoder members helped identify bad ideas and generate better approaches early in the development cycle. When the client introduced a business problem to the community, members asked hundreds of questions. Nic Perez, a former technical director at AOL, explained that the community's questions "gave us insights into problems I didn't even really know I had" and "saved us money by doing all of those questions upfront." Using online forums, clients answered questions for all competitors only once, avoiding repeated efforts. In some cases, clients scrapped product ideas entirely after the community raised concerns about the product's likely success or usability in the marketplace.

TopCoder's contest-based development system consistently produced highly creative ideas and solutions. According to Darren Smith, a solution architect for the e-commerce division at Ferguson Enterprises, North America's largest plumbing supplies wholesaler and distributor, "The community comes back with many options. It really has surprised us. You never know what you are you going to get. The creative side allows us to go to the marketing management team and say, 'We could do X, Y, and Z that we may not have previously considered.' They're adding value to our business because they bring us solutions that quite frankly we may not have considered or were not resourced to deliver."

Superior Quality, Cost, Speed, and Flexibility Clients praised TopCoder's rigorous evaluation and documentation process for being well above industry standards. Reflecting on his experience working on the Google Talk interface to AOL Instant Messenger, Perez stated that TopCoder and its community had a strong desire to deliver bug-free code and that even the most complex systems always had fewer than 100 identified bugs. According to Perez, the same sized projects, developed internally, at AOL would have had five to eight times that number of bugs.

Another TopCoder client, a Web-based startup business, noted that it would have had to pay $350,000 to a large IT consulting firm, $200,000 to a small IT consulting firm, or $80,000 to individual contractors to build the company's website. Using TopCoder, the client only spent $35,000. This same client proclaimed: "At $35,000, it's priceless. There is no other game out there." A different client noted that based on its experience working with almost every type of software development company, TopCoder charged approximately half of the fee of a large, tier-one IT consulting firm.

Using the community for parallel problem solving, TopCoder marketed itself as faster than other software development shops. This was true for back-end bug races and system checks, as TopCoder took 72 hours to complete the same bug testing that a traditional development firm finished in 10 business days. However, for other steps in the software development process, reports on speed were mixed. Some clients said that TopCoder worked at about the same speed as a large IT consulting firm, while others lauded TopCoder for speed of completion.

Especially appealing to clients was TopCoder's ability to supply flexible software development capacity. In particular, a TopCoder client could expand or reduce its business requirements and development capabilities without having to hire or fire programmers. According to one client, a basic in-house computer programmer cost $120,000 a year, after accounting for benefits, sick time, and vacation. Working with TopCoder, clients did not have to spend as much on employees' benefits and downtime.

Concerns

Although CIOs were impressed by TopCoder's technical capabilities and cost-saving potential, many often had initial reservations about working with TopCoder's unusual software development model.

Intellectual Property (IP) and Security According to Ira Heffan, TopCoder's chief legal counsel, "For new clients unfamiliar with TopCoder's model, IP and security concerns can be an initial point of resistance. Until they understand the documentation and processes we have in place with the community members, they see IP and security as potential barriers to working with a community." For example, some clients were apprehensive that a TopCoder community member might divulge proprietary ideas, business plans, or operations to their competitors. In addition, some clients worried that once a component became an integral part of their IT systems, the community member who built the component might attempt to prohibit its use or ask the client to pay considerable royalties. Lastly, some clients were concerned that a solution submitted by a community member could be stolen, copyrighted, or taken from open-source software projects, thus potentially opening the door for intellectual property disputes.

TopCoder had in place a number of initiatives targeted at addressing these concerns and reducing the risk level for clients, and also took steps to communicate its processes. To ease clients' intellectual property and security concerns, TopCoder produced a white paper that detailed confidentiality policies, intellectual property assignment rules, and TopCoder's modular approach to software development. In addition, TopCoder allowed clients to keep their company names anonymous during competitions and helped clients generate test data sets to avoid the exposure of sensitive information. At the client's request, before a community member was allowed entry into a competition, all competitors could be required to sign a standard competition confidentiality agreement.

The peer-review process was another means to ensure code security and quality. Peer reviewers were selected and vetted by TopCoder employees based on their superior performance on prior competitions. TopCoder clients also had the option of running testing competitions at the back end of software production, serving as an additional means of checking code security and quality. TopCoder's compartmentalized software development process also made it difficult for a single competitor to insert harmful code into a program, since individual contests only addressed one small piece of the overall program.

Cultural Change Many clients realized that working with TopCoder would be difficult culturally for their company. In particular, CIOs believed that internal employees would view TopCoder as a threat to their job security. One new client observed: "TopCoder is a CIO's dream but a programmer's worst nightmare. I fully expect that if this goes well and if my programmers see good quality work coming out of TopCoder, fear will quickly explode throughout the building." Although using TopCoder could help a company scale and reduce the programming staff costs, companies still had to retain the "big thinking" people—the employees who could guide the

TopCoder development process. The managers at TopCoder clients also had to adjust to a perceived loss of control over the software development process. Smith commented: "We set the competitions, but they manage the whole process. Our project management group works with the TopCoder manager to ensure delivery according to pre-determined service level agreements (SLAs)." Additionally, some clients found a few community members to be pushy and rude during pre-competition question-and-answer sessions.

Coding Challenges Even if TopCoder did all of a company's internal development work, the company still needed to have an internal staff to integrate the deliverable into the client's existing systems, review the code for security issues, and adjust and fix code as systems changed over time. For example, Smith's team at Ferguson spent a significant amount of time inspecting, testing, and processing TopCoder's work to make sure that there were no security threats or bugs. In some contests, TopCoder clients also spent time evaluating ideas and approaches from multiple winning solutions.

Another ongoing issue for clients was finding the right types of problems and providing the appropriate amount of problem detail for the TopCoder community. As Moore described it, "You want neither too much nor too little detail. You do not want to quell innovation but also want a solution that makes sense in your system." Clients discovered that contest participation decreased if they were unclear about what problems they wanted to solve or presented problems that were too complex or vast in scope; in those cases, the TopCoder community struggled to produce an acceptable solution. Clients also found that community members worked best when contests lasted less than two weeks. If projects took too long to complete, contestants would lose interest and not make submissions.

Managing TopCoder

The Supply Side

A management job at TopCoder was unique. Along with supervising internal TopCoder employees, managers at the firm had to oversee a community of over 200,000 members and direct the process of competition-based software development. According to senior vice president George Tsipolitis, the key to success was effective process management: "When you're managing a community, you are no longer managing individuals, you're managing a whole. We can't control individuals. We can only control the process of their participation." Clients and employees alike believed that the sustainable value of the company was dependent on TopCoder's ability to facilitate community participation and foster community growth. Lydon described the risks: "From the beginning, we focused on the community. We knew they could be unforgiving. If you did the wrong thing, you got crucified."

Attraction To run many competitions simultaneously and produce solutions for many clients at the same time, TopCoder needed to have access to a critical mass of talent and coding capacity. TopCoder's primary means of attracting new members into the community was the appeal and challenge of the algorithm contests. In addition, TopCoder occasionally advertised its online competition platform by paying for Google keyword searches using terms such as "design contests." A third mechanism for attracting talent was "member development days." Organized by a small team of TopCoder employees, member development days were held at Chinese and other international universities. At a member development day, a student representative would post signs around the school and explain the TopCoder system. A primary goal of these member development

days was to encourage participation in the higher-revenue-producing development and design contests. During one member development day in China, TopCoder registered over one thousand new community members. Bourdon noted that TopCoder had achieved critical mass once it crossed the 200,000 member threshold, as there were now many members with deep and narrow skills over a range of software development challenges (See **Exhibit 7** for the number of participants by contest type).

Norms As the community grew, TopCoder paid close attention to establishing community norms. As contest administrator, the company had to maintain the highest standards of contest integrity, fairness, transparency, and quality. For example, TopCoder personnel strictly monitored competitions and tolerated no form of cheating. Community members who peeked at other competitors' solutions, shared ideas during competition, or used unauthorized code were immediately eliminated from the contest. Often they were kicked out of the community entirely.

If any uncertainty or disagreement arose about which competitor won a particular contest, TopCoder would spend extra money to re-run the competition. Another part of contest integrity, Tsipolitis explained, was TopCoder's emphasis on maintaining consistency of rules and procedures: "The second that participants can't figure out how to win, they'll stop participating. So we can't change the rules of a competition mid-stream." TopCoder also guaranteed complete contest transparency by storing all contest and competitor statistics, peer reviews, and solutions in a data warehouse. The data were publicly available to registered community members, accessed via the TopCoder website.

Integrity and fairness also extended to TopCoder's corporate motivations and community compensation philosophy. In particular, TopCoder was up-front with the community over its intention to make money. When TopCoder made a decision to change corporate direction or competition procedures, Hughes posted the information in the forums and explained the business reasons behind his decisions. Hughes also believed that, since the company benefited from the community's hard work, adequately compensating community members was essential.

Governance Although TopCoder executives were responsible for final decisions, they frequently incorporated community members' views into the process. Lydon explained: "We treat the community as the driver for everything we do. If we have enough dissent from members, we always take that into account. The problem is that when we don't know what to do, our members will also be split." Community member Gasper shared a similar perspective: "TopCoder tends to push out ideas into the forums to get feedback. Seventy-five percent of the time, they listen to the community. But TopCoder also has its own business interests to consider. Sometimes the community and business interests don't line up."

Similarly, if competitors were unhappy with a peer-review scoring outcome, TopCoder allowed them to appeal the decision. Over 90% of contests featured at least one appeal. If a member appealed, peer reviewers had to provide specific reasons why the appeal was accepted or rejected. If disagreement remained between contestant and reviewer, TopCoder employees often investigated. Contestants could also appeal directly and privately to TopCoder personnel or post complaints publicly on the TopCoder forums.

TopCoder managers inevitably made decisions that sometimes disturbed and upset the TopCoder community. For example, facing a very difficult economic environment in the summer and fall of 2008, TopCoder reduced the contest prize amounts, cut payments to peer reviewers, and reduced the number of algorithm competitions. During this period, some TopCoder competitors left the community entirely and others dramatically reduced their participation levels. Gasper argued that

the payment cuts also led to many superficial reviews because the best reviewers were no longer doing the work, which then required additional cycles to achieve acceptable quality.

Resource Allocation Another part of the TopCoder managerial role was allocating community resources and controlling contest participation. Lydon explained, "We have to figure out how to distribute the number of people who want to participate across the number of contests that have to be solved." Unlike a typical software development firm, TopCoder could not assign specific people to a task or project. As participation manager, TopCoder's goal was to minimize the costs of evaluation, stimulate effort through competition, and get at least one solution that was acceptable to the client. To achieve the ideal number of submissions and participants, TopCoder adjusted the prize amount, the duration and timing of the contest, the number of other contests running concurrently, and the problem's complexity and scope. When deciding on the ideal number of competitors, TopCoder also considered contestants' reliability. As a last resort, TopCoder employees reached out directly to individual community members if other methods did not lead to the desired participation levels.

Although TopCoder managers could pull many levers to influence contest participation, they believed it was important not to act like the community's boss. Hughes explained his community management philosophy: "We don't own this community. We want people to be here when they want to be here. You are just going to get much better results when you let people do what they really want to do."

Retention At the same time, TopCoder executives worked to retain community members and encourage future contest participation. At least one client raised concerns in this area: "I think that communities are fickle. Community members could start to ask, why do I need them? For example, what happens if an imitator comes along and offers twice the prize amount?" To avoid such problems, TopCoder tried to supply community members with consistent work streams and prize money. TopCoder also encouraged community members to engage in the community as much as possible by dedicating significant resources to facilitating forum discussions and inviting contestants to participate in peer reviews, write problems for contests, and develop TopCoder's internal systems.

TopCoder community members differed on their level of loyalty to the TopCoder community. Paweska, appreciative of all the opportunities TopCoder had provided, reflected, "I have some loyalty. I think it would take a lot for me to leave. Only if there were no projects would I leave." Gasper viewed his position in the community in a different light. "I'm not super loyal to TopCoder or anything," he said. "[For me] to defect, the payment and work would have to outweigh the payment and flexibility I have at TopCoder."

The Demand Side

Platform Managers The other side of management at TopCoder was guiding clients through the contest-based software development process. This was the responsibility of the company's platform managers, whose job was to induce the appropriate amount of community participation, make suggestions for contest prize amounts, gather feedback between contests, and provide project status updates to clients. Before starting the next step in the game plan, platform managers also adjusted contest requirements based on the work already completed. Once the product was delivered to the client, TopCoder platform managers were required to act in a support and service role. If there was a technical problem with a solution, the platform manager often contacted the community members who developed the component and worked with the community members to fix the issue.

Most enterprise-level clients believed the platform manager was pivotal to a project's success. At a large client like LendingTree, the platform manager was on site three to four days a week, conducting daily meetings with the internal teams. A large part of the platform manager's role was managing client expectations and serving as a sounding board for client concerns. At the back end of projects, although the community often assembled a project's components through competition, the platform manager was also an expert at combining the small software pieces. The component integration role saved the client hours of work trying to figure out how all the pieces fit together. At Ferguson, Smith considered the TopCoder personnel working on site to be an integral part of his team.

TopCoder Direct However, each platform manager added to TopCoder's overhead costs and narrowed profit margins. As of 2009, a typical platform manager at TopCoder cost $100,000 a year including benefits. To Tsipolitis, the platform manager's time was not always well spent. "Our project managers spend a lot of time babysitting," he said.

To avoid a potentially large increase in expenses as TopCoder added clients and projects, Hughes came up with the concept of "TopCoder Direct," in which the client used the company's platform with little to no intervention from its employees. Under this self-service model, platform managers would educate clients on how to use the TopCoder platform to manage the contest-based software development process themselves. Hughes envisioned shifting the platform manager's responsibilities to an experienced community member or an external consultant familiar with TopCoder's platform—someone who would serve as a "co-pilot" to assist the client staff. With co-pilots taking the role of platform managers, Hughes estimated that the platform manager's weekly time on a project would shrink from 40 hours to two, thus saving the client and TopCoder considerable time and money.

The Future

As of December 2009, no competitors had elected to copy TopCoder's business model by offering full-service software development through a competition-based approach. Instead, companies such as RentACoder, Elance, and oDesk served as online liaisons between clients and freelance software developers. Unlike TopCoder, whose clients only paid for solutions, clients at these firms used a "buy talent" approach: they selected one or more programmers to solve their problem. More similar to TopCoder, uTest used crowdsourcing to find bugs and check the functional usability of web, mobile, desktop, and gaming applications, but did not engage in software development. According to Hughes, this lack of direct competition reflected the technical difficulties and costs associated with building a full-fledged community and platform.

After a significant downturn in the global economy in 2008 and 2009, Hughes believed that TopCoder was primed for growth. Sales staff were forecasting aggressive targets for the volume of competitions and revenues, and several strategic partnerships were under consideration. However, significant challenges and uncertainty remained. In particular, Hughes wondered whether the community, as well as the company, could grow to meet increasing demand.

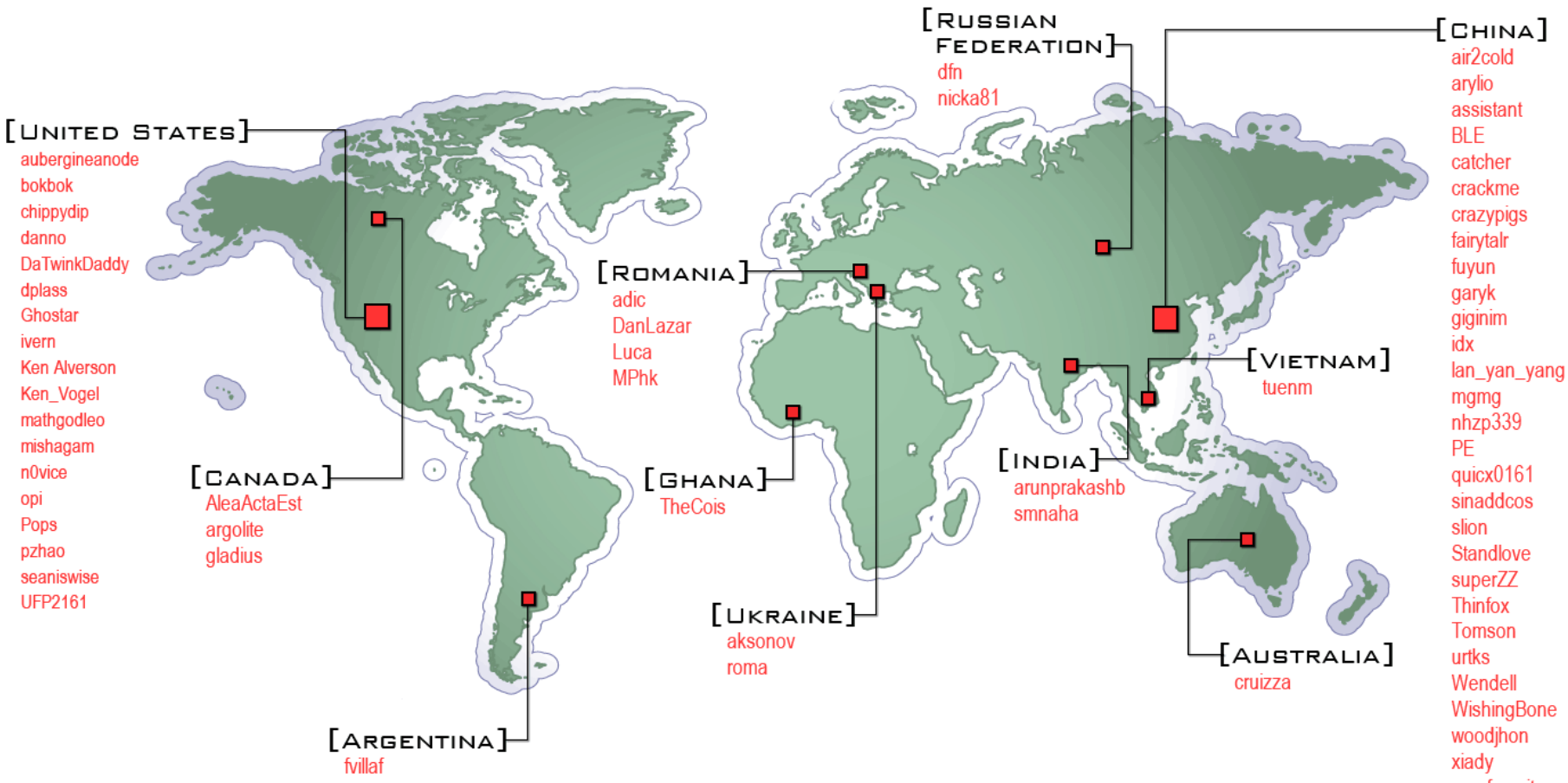
Stakeholders had divergent views. Mike Morris, vice president of sales, saw unlimited potential: "If sales grow at a linear rate, membership grows at an exponential rate. The supply of community members is not going to limit growth. If you throw enough money out there, you will get enough programmers." Community members Paweska and Wu agreed that offering more money per contest would increase participation among existing members. Paweska also believed, however, that holding many more contests than usual in a given week would result in inexperienced competitors competing actively for the prizes, possibly reducing code quality. Furthermore, Lydon noted that that during TopCoder's last large scale-up in 2007, review quality suffered during a transitional period of a few

months. As more contests became available, the usual reviewers wanted to compete in the contests, rather than review them, leaving TopCoder scrambling to find replacements. In addition, a few clients worried that as the number of avenues of competition at TopCoder grew, attracting the same group of competitors would prove much more difficult, reducing contest consistency and continuity, which were especially critical for addressing legacy systems.

Hughes also worried about client service. If the number of TopCoder clients expanded significantly, TopCoder's staff might face increasing difficulties responding to all those clients' questions and concerns. For large clients, expansion might require adding more platform managers, but Rob Hughes, TopCoder's COO, was concerned that too many platform managers might make the firm appear to be like any other large IT consulting company, with the risk of losing its unique business model.

Even if Hughes succeeded at growing TopCoder, he was unsure about the company's competitive position. In particular, Hughes wondered if community members would stick with TopCoder if a new competition-based software development company emerged. What would happen if a company like Accenture started to develop software in the same way as TopCoder? Would the TopCoder community remain intact?

Exhibit 1 TopCoder Members Involved in Creating a Power Pricing System for an Energy Company



Source: Company documents; developed via a TopCoder Studio competition.

Exhibit 2 Number of Clients, Revenue, Number of Platform Managers, and Platform Manager Costs by Quarter

	2007				2008				2009			
	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
Number of Clients	32	32	34	25	24	38	37	47	46	47	38	35
Total Revenue ($MM)	4.66	4.50	3.80	5.35	5.80	5.50	4.85	2.60	2.25	1.92	1.82	2.45
Number of Platform Managers	51	51	44	52	52	46	44	36	20	19	18	16
Cost of Platform Managers ($MM)[a]	1.23	1.24	1.12	1.30	1.36	1.19	1.13	0.91	0.53	0.49	0.46	0.40

Source: Company statistics.

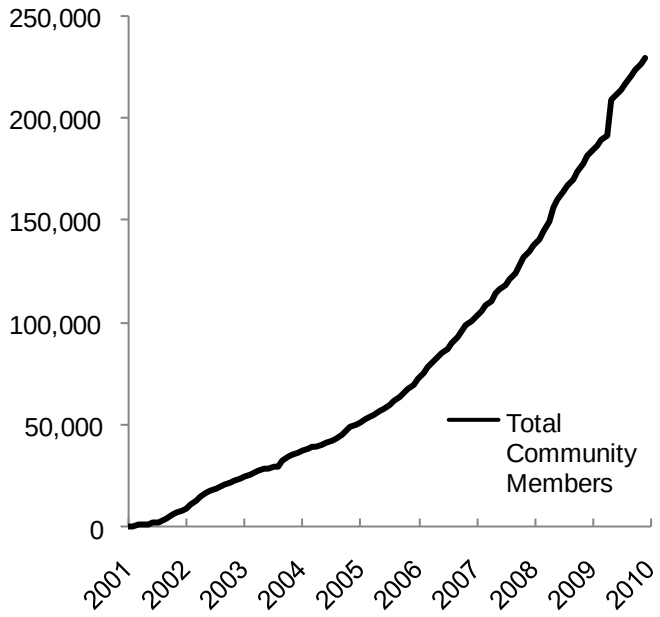
[a] Includes platform managers' salaries, benefits, and other expenses.

Exhibit 3 Sample Game Plan

Phase	*Timeline*																*# of Contests*	*Estimated Costs*
	4/27	5/4	5/11	5/18	5/25	6/1	6/8	6/15	6/22	6/29	7/6	7/13	7/20	7/27	8/3	8/10		
	M T W T F	M T W T F	M T W T F	M T W T F	M T W T F	M T W T F	M T W T F	M T W T F	M T W T F	M T W T F	M T W T F	M T W T F	M T W T F	M T W T F	M T W T F	M T W T F		
Conceptualization																		
Logo - Tournament																	0	$ -
Concept Contest - Tournament	x x x x x																1	$ 4,500.00
Wireframes - Tournament		x x x x x															1	$ 3,000.00
Storyboard - Tournament			x x x	x x x x x	x												1	$ 3,500.00
Application Build																		
System Architecture																		
System Architecture			x x x	x x x x x	x x												1	$ 4,500.00
Component Design					x	x x x x x	x										1	$ 2,300.00
Component Development							x x x	x x x x x									1	$ 2,050.00
Catalog Components																	0	$ -
System Assembly									x x x x	x x x x x	x						1	$ 4,900.00
<Functional> Module																		
Module Specification					x	x x x x x	x x x x										1	$ 4,000.00
Module Architecture								x x x x x	x x x x x								1	$ 4,000.00
Component Design										x x x	x x x x x						3	$ 6,900.00
Component Development												x x x x	x x x x x				3	$ 6,600.00
Catalog Components																	0	$ -
Module Assembly														x x x x	x x x x x		1	$ 4,900.00
Prototype Assembly							x	x x x x x	x x x x								1	$ 4,900.00
Testing																		
Test Scenarios						x x x x x	x x x x x										2	$ 3,900.00
Test Cases								x x x	x x x x x	x x							2	$ 3,900.00
Bug Hunt														x x x	x x x x x		1	$ 2,750.00
Deployment																		
Deployment						x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	20	$ 4,000.00
Updates																		
Bug Races		x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	x x x x x	30	$ 4,200.00
																	Total	$ 74,800.00

Source: Company documents.

Exhibit 4 Community Growth



Source: Company statistics.

Exhibit 5 Average Contest Registration, Submission, and Prize Amount for Client Contests in 2008 and 2009

Contest Type	2008			2009		
	Number of Registrants per Contest	Submissions per Contest	Prize Amount per Contest[a]	Number of Registrants per Contest	Submissions per Contest	Prize Amount per Contest[a]
Conceptualization	n/a[b]	n/a[b]	n/a[b]	17.57	3.60	$1,314
Specification	n/a[b]	n/a[b]	n/a[b]	14.20	1.94	$1,017
Architecture	16.3	1.64	$1,590	19.34	1.75	$1,095
Component Design	9.83	2.85	$899	16.26	1.94	$559
Component Development	15.4	2.69	$733	25.59	2.56	$465
Assembly	16.09	1.12	$1,628	18.38	1.18	$913
Studio	27.55	14.57	$795	27.57	20.04	$1,015

Source: Company statistics.

[a] Prize per contest – Prize for first and second places and reserve for Digital Run.

[b] n/a – Data not available for most of 2008.

Exhibit 6 Example Community Profile

Statistics Member Profile

argolite

Algorithm Rating:	1263
Conceptualization Rating:	not rated
Specification Rating:	not rated
Architecture Rating:	1317
Design Rating:	1529
Development Rating:	1213
Assembly Rating:	not rated
Test Suites Rating:	not rated
Marathon Matches Rating:	not rated
Total Earnings:	$815,272.92
Member Since:	06.04.2002
Country:	Canada

[Send a message]
[Forum post history]
[Achievements]

Quote:

""The greatest trick the Devil ever pulled was convincing the world that he didn't exist." - Keyser Söze"

Algorithm | Conceptualization | Specification | Architecture | **Design** | Development | Assembly | Test Suites | Marathon Matches

Design Competitions

Rating: 1529

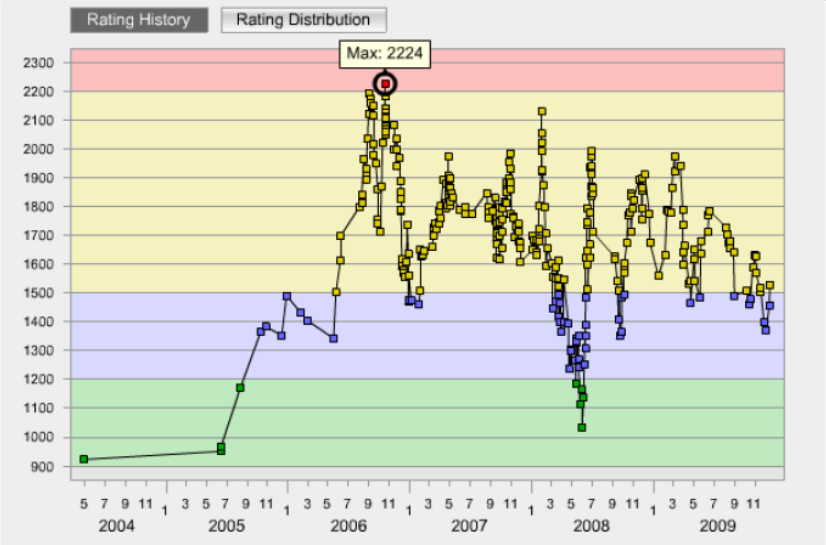
[competition history]
[current contests]
[reliability detail]

Percentile:	67.2414
Rank:	19 of 58
Country Rank:	1 of 4
Volatility:	352
Competitions:	345
Reliability:	100%
Maximum Rating:	2224
Minimum Rating:	922

Submission Details

	Java	.NET	Total
Inquiries	204	140	345
Submissions	192	128	321
Submission Rate	94.12%	91.43%	93.04%
Passed Screening	192	128	321
Screening Success Rate	100.00%	100.00%	100.00%
Passed Review	191	127	319
Review Success Rate	99.48%	99.22%	99.38%
Appeals*	1375	724	2110
Appeal Success Rate*	53.09%	48.07%	51.33%
Maximum Score	99.27	98.50	99.27
Minimum Score	47.33	66.77	47.33
Average Score	91.78	91.61	91.73
Average Placement	1.54	1.38	1.48
Wins	115	87	202
Win Percentage	59.90%	67.97%	62.93%

* only includes appeals from projects posted on or after March 16, 2006 (TopCoder did not previously collect the relevant data)



Source: http://www.topcoder.com/tc?module=MemberProfile&cr=287614, accessed December 23, 2009.

Exhibit 7 Number of Unique Participants by Contest Type per Year and Month/Total Number of Official Contests per Year

	2005			2006			2007			2008			2009		
Contest Type	Average Submitters per Month	per Year	Contests per Year	Average Submitters per Month	per Year	Contests per Year	Average Submitters per Month	per Year	Contests per Year	Average Submitters per Month	per Year	Contests per Year	Average Submitters per Month	per Year	Contests per Year
Algorithm Contests															
Single Round[a]	1,319	5,287	81	1,930	7,525	105	2,638	8,994	113	2,945	10,433	66	2,558	9,616	51
Marathon Match[b]	500[c]	500[c]	1[c]	273	1,532	20	281	1,588	30	253	1,621	29	314	2,150	35
Client Software Development Contests															
Conceptualization										7[c]	11[c]	9[c]	6	34	70
Specification										5[c]	7[c]	16[c]	6	32	71
Architecture							1[c]	4[c]	4[c]	4	29	65	9	36	145
Component Design	33	157	362	45	225	615	61	243	698	38	144	488	20	93	300
Component Dev.	58	316	287	88	451	484	135	605	780	91	434	733	41	204	337
Assembly							7	47	86	10	59	191	20	104	416
Design Contests															
Studio				65[c]	223[c]	17[c]	81	453	118	66	279	451	102	429	456
Total[d]	1,370	5,565	730	2,146	8,517	1,224	2,867	10,072	1,825	3,198	11,487	2,023	2,911	11,122	1,881

Source: Company statistics.

[a] 75-minute programming contest.

[b] Programming contests that run from 3–30 days.

[c] Partial year data—contest track officially did not officially start until the middle or end of the year.

[d] Represents the unique number of participants during a given time period. Columns are not additive.