



© Tadeusz Ibram | Dreamstime.com

# Where in the World Is Carmen Sandiego (and Is She a Software Engineer)?

Phillip A. Laplante, *Penn State*

**W**hen my kids were young, there was a cartoon show called *Where in the World Is Carmen Sandiego?* Popular from the early to mid-1990s, the show was centered around Carmen Sandiego—an evil, globe-trotting, ex-secret agent. The franchise began as a computer game published by Broderbund Software and was intended to teach kids about geography. Sandiego and her associated characters later spawned books, videos, licensed merchandise, and television shows that aired in several countries.

Recent conversations pertaining to licensure of software engineers in the US and offshoring had me thinking about Carmen again. I wondered how well she would have fared if her secret agent cover was as a software guru. Would she be able to pass as a software engineer in several different countries—for example, in the US, India, Malaysia, Argentina, and Ireland? Are there differences in how software engineering is practiced in different cultures that are so profound that even a super spy, expert at mimicry and disguise, couldn't fool her colleagues?

The answers to these questions have important implications for the transnational practice of software engineering, for outsourcing and offshoring policy and strategy, and even for national security.

## How Culture Effects Behavior

In the book *Outliers*, Malcolm Gladwell recounts how cultural mismatches and a resistance to challenge authority in the communications protocol used between international airline pilots and air traffic controllers led to catastrophe in 1997.<sup>1</sup> He describes how the reluctance of Korean Airlines officers to protest the instructions from an American air traffic controller at Guam airport led to a crash that killed 228 people.

Others have written about the problems of managing cultural differences in transnational IT projects,<sup>2,3</sup> and there's plenty of work on how to apply process models such as the Capability Maturity Model in multinational settings.<sup>4</sup> But the question I'm addressing here is in the actual practices of software engineering—not the management of it. In other words, do people practice

software engineering differently because of cultural differences? Should a professional exam testing minimum competency (or proficiency) in software engineering in the US be the same as in any other country?

In the early 1970s, sociologist Geert Hofstede studied a large data set pertaining to IBM employee values that had been collected across worldwide sites between 1967 and 1973. Hofstede found that these values varied significantly depending on the site at which the employees worked. Hofstede concluded that there were four dimensions of social norms along which cultural differences between countries could be perceived: the power distance index (PDI), individualism (IDV), masculinity (MAS), and uncertainty avoidance (UAI). He later added a fifth dimension, long-term orientation (LTO).

I'll describe each of these briefly in the context of software engineering. The IT profession and software engineering are good candidates for examination under the lights of Hofstede's metrics because of the likely collaboration between customers, vendors, and colleagues across several

countries; the frequent use of outsourcing and offshoring; and the use of third-party and externally furnished components of international origin.

### **Power Distance Index**

PDI indicates the relative acceptance of equal power distribution across different groups or classes. A high power distance indicates an unwillingness to challenge those at a higher authority.

A low number corresponds to the willingness to challenge authority—an engineer who questions a project manager trying to ship a product without thorough testing, or a requirements engineer asking a high-powered stakeholder to clarify a critical requirement or raising a concern about a serious problem hidden from inspectors.

### **Individualism**

IDV rates the extent to which the culture values the individual over the collective. Individualistic societies value self preservation over sacrifice for the greater good.

A high IDV number indicates a strong individualistic tendency and the propensity to take credit for someone else's work or to avoid blame. So, highly individualistic software engineering activities, such as testing or coding, might be more attractive in countries with a high IDV.

### **Masculinity**

MAS describes the extent to which the differences in societal roles vary between genders. In high-MAS cultures, there's a large separation between conventional male and female roles; it would be unusual or even forbidden to have a woman in a high-ranking position.

In such cultures, could engineers sufficiently represent the female perspective on, say, user

interfaces or operational features for automobiles, medical equipment, and potentially dangerous consumer products?

### **Uncertainty Avoidance**

UAV measures a society's overall willingness to accept uncertainty,

tolerate differences in opinion, and require structure in organizations. A high uncertainty metric indicates that a culture doesn't tolerate strong differences in opinion and prefers structured situations.

In such an environment, would the practices of software requirements engineering or testing, where challenges to authority must be made, differ from practices in a low-UAV setting?

### **Long-Term Orientation**

LTO refers to a culture's attitude toward the following: long-range planning, saving, patience in recovery from short-term setbacks, preservation of reputation, and respect for tradition. A high number indicates that a long-term view is highly valued in the society.<sup>5</sup>

Are software engineers in low-LTO countries more likely to favor a code-and-fix approach to formal methods? Are software engineers in high-LTO countries more likely to favor spending more time on requirements engineering and less on testing?

### **Is Software Engineering the Same Everywhere?**

I'm part of task force charged with developing a professional licensure examination for software

engineering to measure minimum competency.

Texas is the only state in the US to currently require licensure for software engineers working on systems that affect public safety and welfare, but nine other states have signaled that they also will

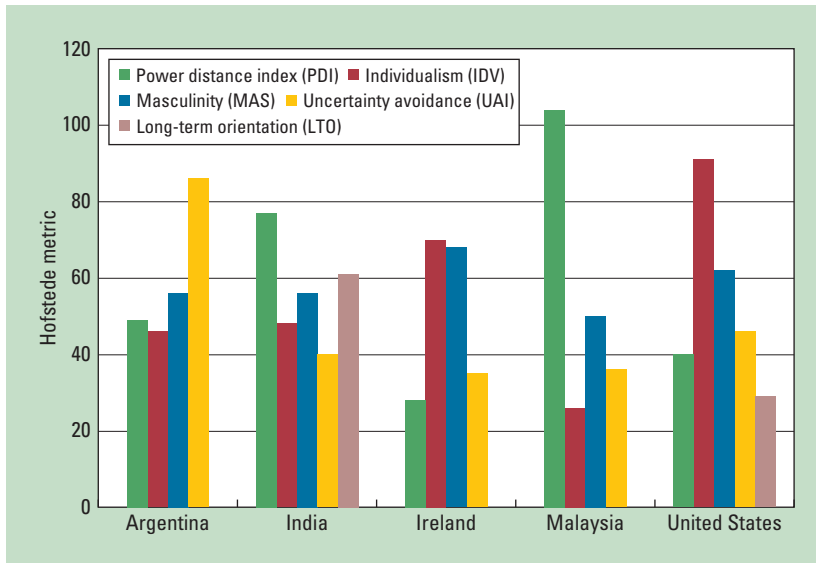
## **The question arises ... are there cultural differences that impact the practice and possibly the products of software engineering activities?**

license software engineers. I expect every state to eventually follow. Furthermore, this licensure exam activity has drawn the attention of officials from several other nations. My understanding is that these countries are also seeking to establish some sort of licensure process for their software engineers.

So the question arises, is the practice of software engineering the same in all countries, or are there cultural differences that impact the practice and possibly the products of software engineering activities? Will the work that we do in creating a software licensure exam be applicable in other countries?

Hofstede's metrics aren't known for every country, but metrics are available for many countries where you'd expect software development to occur regularly. Since the original study, Hofstede has updated and expanded his work, and this data is readily available on his website ([www.geert-hofstede.com](http://www.geert-hofstede.com)). Using this data, I gathered the Hofstede metrics profiles for five countries (see Figure 1).

In reviewing the figure, note the high UAI number in Argentina (86). Do you think that a requirements engineer there would be more likely to accept a vague software requirement for a fail-safe mechanism on an implanted



**Figure 1. Hofstede metrics for five countries. Long-term orientation data was available only for the US and India. (Data gathered from [www.geert-hofstede.com](http://www.geert-hofstede.com).)**

insulin pump? I would expect formal methods and time spent on upfront design and analysis to be valued higher in Argentina than in the US.

Would software engineers in Malaysia (PDI = 104) use fewer techniques (such as reviews) that require higher management participation than in Ireland (PDI = 28)? How widely are group reviews (and the concomitant criticism) used in the US, where individualism is high (IDV = 91) versus in India (IDV = 48)?

For the five metrics, using only the measures “high” versus “low,” there are 32 different ways that there could be mismatches in process or perception of various software engineering practices. So how would Carmen fare as she traveled undercover around the world as a licensed professional software engineer from the US?

As a “macho” secret agent, she might have a tough time deferring to her managers in Ireland, where MAS is fairly high at 68. Her strong individualism might make it tough for her to be part of

a requirements engineering team in Malaysia, where the IDV is 26. Being assertive, she wouldn’t do well as a tester in India, where the PDI is 77. And her low uncertainty avoidance (she’s a spy, so she thrives on uncertainty) would make it hard for her to pass undetected in Argentina, where the UAI is 86.

### Culture and the Practice of Software Engineering

While there have been many studies on global software development, no one knows if the practice of software engineering is the same in all countries, and no one knows if there could be a one-size-fits-all competency test. Interaction in virtual worlds might provide a mechanism for testing the boundaries of cultural differences in software development in a safe, low-cost way. Maybe we should create a Carmen Sandiego character and put her on a whirlwind tour of Second Life’s various islands, where she can work on software projects with avatars from other countries.

To begin the development process for the licensure exam, our committee will develop a survey instrument to assess the professional activities and knowledge areas for software engineers working on systems that affect public safety and welfare. We plan on surveying hundreds of practicing software engineers, including some from outside the US, to determine the relevant knowledge areas and practices for the exam.

At a recent talk to the New York City IEEE Computer Society section, I discussed the state of software engineering licensure efforts in the US and the road forward. A gentleman at the meeting asked me if a survey of software engineering professionals on the practices and knowledge of the profession taken in the US would apply to other countries. I said I didn’t think it would, but we need to wait to see the data from our survey. Since we’ll know each respondent’s country of practice (though not origin), we’ll be getting some insight as to whether there really are national differences in the practice of software engineering. We hope to publish this work in the future.

Of course, each nation is culturally diverse to varying degrees due to influences of immigrants and long-term visitors. But Hofstede acknowledges this fact in his studies. Culture is the persistent collection of norms after all outside influences have had their effect—culture is supposed to be more enduring.

Whatever the influences of culture, the question still remains, “Is the practice of software engineering the same in every country?”

I don't know, but I'm trying to find out. **IT**

## References

1. M. Gladwell, *Outliers: The Story of Success*, Little, Brown and Company, 2008.
2. S. Deshpande et al., "Culture in Global Software Development—A Weakness or Strength?" *Proc. 5th Int'l Conf. Global Software Eng. (ICGSE 10)*, IEEE CS Press, 2010, pp. 67–76.
3. K. Jablolkow and M. Myer, "Managing Cognitive and Cultural Diversity in Global IT Teams," *Proc. 5th Int'l Conf. Global Software Eng. (ICGSE 10)*, IEEE CS Press, 2010, pp. 77–86.
4. C. Ebert and P. De Neve, "Surviving Global Software Development," *IEEE Software*, Mar./Apr. 2001, pp. 62–69.
5. G. Hofstede, *Culture's Consequences: Comparing Values, Behaviors, Institutions and Organizations Across Nations*, Sage Publications, 2001.

**Phillip A. Laplante** is professor of software engineering at Pennsylvania State University. His research interests include software project management, software testing, requirements engineering, and cyber security. Laplante has a PhD from the Stevens Institute of Technology and is a licensed professional engineer in Pennsylvania and a CSDP. He's a fellow of IEEE and a member of the IEEE Computer Society's Board of Governors. Contact him at [plaplante@psu.edu](mailto:plaplante@psu.edu).



The magazine of computational tools and methods.

MEMBERS \$49  
for print and online

STUDENTS \$25  
for print and online

[www.computer.org/cise](http://www.computer.org/cise)

<http://cise.aip.org>

CISE addresses large computational problems by sharing

- » efficient algorithms
- » system software
- » computer architecture

AIP

IEEE

IEEE  
computer  
society

**cn** Selected CS articles and columns are available for free at <http://ComputingNow.computer.org>.

This article was featured in

# computing **now**

ACCESS | DISCOVER | ENGAGE

For access to more content from the IEEE Computer Society,  
see [computingnow.computer.org](http://computingnow.computer.org).



**IEEE**

IEEE  **computer society**

Top articles, podcasts, and more.



[computingnow.computer.org](http://computingnow.computer.org)