The Need to Play Together: The Case for an Open-Source Simulation for Interagency Coordination

Ronald (Skip) Cole

Senior Program Officer
United States Institute of Peace
Washington, D.C., U.S.A.
e-mail: rcole@usip.org

Skip Cole is a member of the Education and Training Center of the United States Institute of Peace. He helps deliver important instructional content developed at USIP to the world via technology. Cole has extensive experience in the field of simulation, both in terms of content and technology. He has developed molecular dynamics simulations, and developed simulations of business and business decision-making processes. He is a Sun Certified Enterprise Architect, Programmer and Developer. He is also an Oracle Certified Database Associate. He was a nuclear reactor operator in the Navy, has worked for a number of private sector firms (including Arthur Andersen) and has teaching experience. He has a B.S. in biophysics from the University of Connecticut summa cum laude, a Master's in biophysics and computational biology from the University of Illinois, and a Master's in global management from the University of Phoenix.

ABSTRACT

There exist widely acknowledged problems with interagency coordination. We do not know where these coordination efforts will lead. Frequently discussed is a 'Goldwater Niccols' like act for the various civilian agencies that have to work together in complex operations. Whatever solution is decided upon, it will involved improved training, and will probably involve the creation of cross-functional multi-agency teams.

Simulation is the next frontier in training. Many things that were learned via rote memorization, lectures, videos, and even via on the job training and experience are now being learned via simulations. It is only natural that the training that will be required for interagency coordination will involve simulations.

While it may be most natural path to follow – teaching interagency coordination via simulation – funding and ownership of such a software simulation will be problematic. If any one agency funds it, then it will own it. The problems of territory and coordinated funding that plague joint interagency actions will exist in the creation of the software that could help them work together.

Fortunately this problem may be circumvented by another trend that is on the rise: open source software. Open source software production allows all parties to jointly contribute, and to have full access to the underlying code. The open source model can, and we predict will, provide a solution that will help the various organizations of the U.S. Government (and even other actors such as members of NGOs) work together better during complex operations.

INTRODUCTION

Currently we are working to develop a simulation tool to help teach lessons learned from post conflict reconstruction experiences. It is possible that this tool will be useful in helping to develop cross-functional multi-agency teams before these teams are called into real action. At the Cornwallis XII meeting, we discussed our efforts and received feedback on what it will take for this tool to be used in the above fashion.

This paper builds the case that not only is the creation of such software achievable and desirable, it is also inevitable. To effectively coordinate during complex operations, people need to experience working together as teams before things get serious. Anything less will lead to ad hoc and suboptimal performance. Teams with geographically separated members (often emergent organizations that only come together for a crisis such as Katrina) still need to spend this time forming. Simulated, or virtual, environments can provide a cost effective way to do this. The huge need for shared simulation training will eventually drive the shared software platform into being.

The constraints on such a software system will be many – it will need to be many different things to different people. For some it will need to teach strategic lessons, for others operational lessons, and for still others tactical lessons. For everyone it will need to allow them to be able to see their role in a very large and very complex operation, so they can see how their decisions influence the success of the larger mission. It will also be necessary to allow subject matter experts to create scenarios based on their specific area. Allowing for 'user generated content' will be essential.

A software system large enough and flexible enough to handle all of the requirements discussed above cannot be created from separate systems cobbled together on an ad hoc basis. It will require that either a shared standard for separate systems to work together (which does not exist) or one shared piece of software to use to create the simulations (which also does not yet exist.)

Either of these paths look possible at this time, and they are not completely mutually exclusive. But our prediction is that an open source product will help drive this 'market' toward one shared solution. The need for this software is great. There are several commercial ventures working on it right now. But once a sufficiently mature open source solution exists, it reduces economic incentive for competing products and increases economic incentives for products that work with it, and modules that build on top of it. This will be sufficient to 'lock in' one product that can be continuously upgraded.

This paper will introduce the need for shared simulations in interagency coordination, describe the open source software movement, and describe how an open source solution could be catalyzed. We say 'catalyzed' for a reason. Open source projects begin as a small kernel around which many can contribute. To initiate such a solution, we don't need to answer all of the issues, but we do need to create the software framework for doing so and create the means by which people can contribute. If the framework is sufficiently robust, and the invitation to contribute sincere, then they will come.

THE NEED FOR BETTER INTERAGENCY COORDINATION

WHO NEEDS IT?

On July 11, 2007 Karen Hughes, Under Secretary for Public Diplomacy and Public Affairs, made these remarks at Department of Defense Conference on Strategic Communication[1]:

> "I hear around the world that we don't speak as one government – yet whether it's a detainee issue or a secret prison issue or dealing with Iran or Iraq issues, mass audiences around the world don't really care if it's news from DOD or CIA or State Dept. – they hear it as news from America – it's increasingly important that we break down silos and coordinate and communicate in one voice."

Examples of failure for the various agencies to coordinate as well expected abound from recent history in Iraq, Afghanistan and New Orleans, so we will not go into detail on them here. But this point should be made: The agencies are, in general, made up of good people trying to do their job well. Frequent failures to coordinate are not due to bad people but due to bad processes.

HOW MANY AGENCIES?

Even in the best of all possible worlds, where all U.S. Government employees take their charge very seriously and selflessly work for the good of the nation (and not for just the good of their particular agency), the sheer number of agencies involved in a complex operation is daunting.

The "Interagency Management of Complex Crisis Operations Handbook" published by NDU in 2003[2] lists the agencies involved Figure 1):

SIMULATIONS TO THE RESCUE

SIMULATIONS ON THE RISE

There seems to be little need to emphasize that simulation is becoming an important aspect of training. According to a recent article in eSchool News[3]:

> "Computer simulations, which have been used for years as training tools by the military and airlines, are increasingly finding their way into professions such as teaching, policing, sales, and other fields that depend more on interpersonal skills than technical proficiency. These so-called "social simulators" chase an elusive goal of replicating human behavior--but their proponents say they provide a safe environment that can be used any time and are a cost-cutting alternative to face-to-face training."

- Agency for International Development
  - Bureau for Humanitarian Response
- Department of Defense
  - Office of the Secretary of Defense
    - International Security Affairs
    - Special Operations and Humanitarian Affairs
    - Stability Operations Strategy
  - Joint Staff
    - Strategic Plans and Policy and Plans, J-5
    - Operational Plans and Interoperability, J-7
  - Other
    - National Defense University
    - U.S. Army Peacekeeping Institute
- Department of Justice
  - International Criminal Investigative Training Assistance Program
- Department of State
  - Regional Bureaus
  - International Organization Affairs
  - Political-Military Affairs
  - International Narcotics and Law Enforcement
  - Population, Refugee, and Migration
  - Democracy, Human Rights, and Labor
  - Economic and Business Affairs
  - USUN-New York and Washington Office
  - Foreign Service Institute
  - Department of Transportation
  - U.S. Coast Guard
- Department of Treasury
  - Office of International Affairs
  - Office of Emergency Preparedness
- Director of Central Intelligence
  - National Intelligence Council/Global Issues
  - DI/Office of Transnational Issues DO
- NSC Global Issues and Multilateral Affairs
  - Office of Management and Budget
  - National Security and International Affairs
- U.S. Information Agency

Figure 1: Agencies involved in Interagency Management of Complex Crisis Operations.

On October 19th, 2006, the MacArthur foundation has recently launched a $50,000,000.00 five-year project to study digital learning. According to their press release[4]:

> "This is the first generation to grow up digital – coming of age in a world where computers, the Internet, videogames, and cell phones are common, and where expressing themselves through these tools is the norm," said MacArthur President Jonathan Fanton, who announced the new initiative today. "Given how present these technologies are in their lives, do young people act, think and learn differently today? And what are the implications for education and for society? MacArthur will encourage this discussion, fund research, support innovation, and engage those who can make judgments about these difficult but critical questions."

While the use of simulations in training is on the rise, it has also had a long historical role in helping groups coordinate. According to Dave Davis, director of George Mason

University's Peace Operation Program, simulations lead the way in helping the U.S. Military coordinate its activities[5].

A SIMULATION TO MAKE A 'TEAM OF CHAMPIONS'

Modern research, and common experience, tells us that teams need time to learn how to work together. Knowing the capabilities of other team members and developing trust are not things that happen immediately. A typical model of team development forwarded by Bruce Tuckman in 1965 was that of forming, storming, norming and performing. The reality is that good teams don't just come together over night.

Interagency cross functional teams will have to develop by simulation training over extended periods of time to be prepared for the complex operations they may face. Simulation centers can be used to help them in preparation, but since the participants are spread out geographically, the drive will be for the participants to be able to participate from their own location. The typical computer desktop can provide a sufficiently rich environment to allow for such simulations delivered over the Internet. So a training simulation that runs on people's desktops and does not require them to all be present at the same simulation center (a 'distributed application') will be highly desirable.

To be successful, this will have to have four highly related elements.

1. A game to Play – This is the 'program' that people will see. It will be loaded with scenarios created by users using the next element.

2. A way to create games to play – This is the 'program' used to create scenarios.

3. An online community to discuss and share ideas – without this the program will grow stale and die.

4. A software owner or maintainer – someone needs to create (or accept) program modifications to help it keep growing.

The first three elements are common elements of complex modern software programs. Frequently there are programs, ways to create modules for programs, and an online community of users who ask and answer each other's questions about those programs.

Element 4 is the most interesting. If the shared software to perform simulations is a proprietary package, then it will have an owner. That owner will probably be able to enjoy very large profits. If the package software package is open source it will not have an owner, but a maintainer. The role of 'software maintainer' is an interesting one. Linus Torvalds plays this role for Linux. This is not a role of complete control. Since one has given over the rights to the software to everyone, if the 'software maintainer' does not manage to keep a large

enough of their user community happy, the community can 'fork[1].' When a community forks, it bifurcates into two or more groups.

DISTRIBUTED APPLICATIONS

As previously discussed, a distributed application to allow for extended simulations for people geographically separated will be highly desirable. In software there are two ways for separate computer systems to communicate efficiently. First, they could both be using the same software. When someone sends you a Microsoft Word document, you probably have no problem opening it – since it is highly likely that you also use a version of Microsoft Word. The second way is for both systems to use some agreed upon standard. For example the TCP/IP standard makes most all of the computer-to-computer communication (from email, to chat, to file transfers) done over the Internet now possible.

The military uses a vast number of simulations, most of which have been created by independent vendors. These pieces of software, such as flight simulators and tank trainers are large, complicated and expensive. They can all communicate with one another since the military coordinates its simulations via a precise standard, the High Level Architecture (HLA) standard. This standard, which works well for modeling the 'kinetic phase' of an operation, is not sufficient for modeling more subtle human reactions – such as how a population will react to a coupe[6].

Could the US Government agencies perform shared simulations via an open standard? Possibly. But such a standard does not yet exist, and given the glacial rates at which both standards bodies and bureaucracies work, it is not likely to exist soon. It is also an open question as to what such a standard would look like. Standardizing sociological phenomena is far more complex than standardizing physical phenomena. For this reason, we believe that the path most likely taken will be the use of a shared software product.

Given the nature of the market we believe that this 'shared software product' will be an open source creation. To explain why we feel this we will first describe what 'open source' is and then go into further detail on how this open source program will probably eventually come to force most competitors out of the market.

THE OPEN SOURCE SOFTWARE MOVEMENT

WHAT DOES 'OPEN SOURCE' MEAN?

This term means different things to different people. But at the simplest level, the term 'open source' means that the source code used to generate a binary executable program is open to anyone for inspection. Computers and humans speak highly different languages. Ultimately

[1] The term 'fork' has a rich history in programming terms. Please suffice it to say that it used here in the sense of 'take a fork in the road.'

all a typical modern computer understands are binary strings. The 'source code' of a program is the human readable set of characters used to generate these strings.

A line of source code may look like this: numberOfDogs = 5;

The compiled binary code may look more like this: 01000101011111100101

Since software, once written, can be copied many times at negligible cost, companies that make a profit selling their programs protect their investment by not allowing others to see their source code[2].

Protecting one's intellectual property rights makes good sense from a business point of view. As Bill Gates wrote in 1976 "Who can afford to do professional work for nothing? What hobbyist can put their three man-years into programming, finding all the bugs, documenting his product, and distribute it for free?[7]" Protecting the source code is a form of protecting one's intellectual property. It is a form of 'security through obscurity.'

Being able to see the source code of a program does not automatically give one the rights to run that software. Copyright still holds, even if the source code has been opened up. But in general if someone has opened up the source code (thus allowing potential competitors seeing it) they have decided to go one of two routes, and will attach the applicable form of copyright notice onto their software. These two basic types are described below.

> Type 1, Viral – You can copy this software and modify it and give it away for free, as long as you acknowledge that you have copied it from us, and any derivative works must also be free.
>
> Type 2, Non-viral – You can copy this software and modify it and give it away for free, as long as you acknowledge that you have copied it from us. Derivative works need not also be free, so you can make money building on top of our work.

This second form of copyright licensing indicates that there can be money made working with open-source software. There are several variant business models[3]. For example the company "Red Hat" provides support for people using Linux.

OPEN SOURCE – APPARENTLY GOOD ENOUGH FOR THE US NAVY

But does open source produce reliable software? Even the military, which requires the highest level of software assurance, has decided that open source software can make the grade. According to a recent GCN article[8]:

---

2 It is possible, given the executable program, to de-compile the binary code but the resulting source is frequently so hard to understand that it would be simpler just to rewrite the software itself. Source code is written by humans for humans to understand. Variables, for example, are given logical names. Once the source has been compiled and de-compiled, all of this information has been lost.

3 We recommend the book "Open Source: A Multidisciplinary Approach" by Moreno Muffatto to describe these business models.

> "Open-source software is now an official option for all information technology systems in the Navy and Marine Corps, according to a guidance memo issued June 5 by the Department of the Navy's Office of the Chief Information Officer.
>
> The Open-Source Guidance memo gives open-source platforms the same status as commercial off-the-shelf and government off-the-shelf software products, allowing Navy IT administrators to evaluate open-source code in acquisitions.
>
> The department "recognizes the importance of [open-source software] to the warfighter and the need to leverage its benefits throughout the [Department of the Navy]," according to the memo issued by Navy CIO Robert Carey."

ASYMMETRIC SOFTWARE WARFARE — THE INEXORABLE DRIVE OF OPEN SOURCE

Software, like literature and much like late night drunken postings on the Internet, once written stays written. It ceases to be useful when one of several things happens:

1. The specific platform (e.g. VAX/VMS) it was written for ceases to be.
2. The problem it addressed is no longer relevant. (For example software to find and correct Y2K bugs is no longer much of a commodity.)
3. A better program replaces it.

Of these three paths to obsolescence, only number 3 is of interest to us here for the foreseeable future. Software platforms have consolidated greatly. We all now use the basic 'Wintel' standard. And the problem we are trying to address, improving interagency coordination, does not seem to be a problem that will go away soon.

Battles for software dominance are commonplace. Rule Number 4 of the 22 Immutable Laws of Marketing[9] is that "In the long run, every market becomes a two-horse race." As markets mature, they tend toward having many players to having just a few key players. This has certainly been true in software, where interoperability and a positive 'network' effect contribute greatly to the consolidation.

Software battles can wage over decades. All players will pour great resources into their program to try to gain market share. A slight lead at one point can be used to gain users, since people purchasing the software want to purchase something that will not die out soon. Thus slight lead can bring about market dominance and great profit. Due to this, these battles can be very bitter and expensive.

At the Cornwallis XII meeting asymmetric warfare and its presentation in the book 'The Sling and the Stone' were discussed at length. In an asymmetric war a very patient insurgency can defeat the political will of a much larger power. The analogy here to software development is that the open source movement operates like an insurgency. (Many of its members are driven ideologically by the strong belief that 'information should be free.')

How can an 'open source' solution drive a large and well-funded competitor out of the field? By convincing them that large profits will not exist for their efforts. Once an open source product exists there is little financial incentive to create a commercial version of the same produce. For example, the online free encyclopedia 'Wikipedia' must be taken into account by anyone writing a business plan to create Encyclopedias.

The above may sound theoretical. Here is a real example taken from the book "The Starfish and the Spider":

> "IBM saw that Linux – the open-source operating system that rivals Microsoft Windows – was gaining traction. Instead of competing with the decentralized market entrants, IBM supported them. It deployed six hundred engineers whose sole job was to contribute to Linux, and it actively supported the development of Apache and Firefox, the open-source browser that competes with Microsoft's Internet Explorer.
>
> IBM's strategy was based in part on the "whoever is my enemy's enemy is my friend" philosophy. That is, "if these programs are hurting Microsoft, our competitor, then let's help them." But it's not just about thwarting competitors. IBM has predicted that open-source is going to win out in the end. The company could spend resources developing competitive products, but chances are they will ultimately lose out. The open-source movement simply has too much momentum.
>
> Rather than try to develop a competitive operating system in-house, IBM supported the development of Linux, then designed and sold hardware and software that was Linux-compatible. IBM is harnessing the collective skill of thousands of engineers working collaboratively worldwide, and at no cost to IBM."

THE CATHEDRAL AND THE BAZAAR

A large part of the mindset difference between the proprietary and open source worlds are captured in the essay "The Cathedral and the Bazaar" by Eric Steven Raymond. In it he describes the creation of Cathedrals as centrally planned, and Bazaars as things that the community seems to bring together, as if by magic. Raymond wrote[10],

> "Linus Torvalds's style of development—release early and often, delegate everything you can, be open to the point of promiscuity—came as a surprise. No quiet, reverent cathedral-building here—rather, the Linux community seemed to resemble a great babbling bazaar of differing agendas and approaches (aptly symbolized by the Linux archive sites, who'd take submissions from anyone) out of which a coherent and stable system could seemingly emerge only by a succession of miracles.
>
> The fact that this bazaar style seemed to work, and work well, came as a distinct shock. As I learned my way around, I worked hard not just at individual projects, but also at trying to understand why the Linux world not

> only didn't fly apart in confusion but seemed to go from strength to strength at a speed barely imaginable to cathedral-builders."

Creating a bazaar is distinctly different than creating a cathedral. The first difference is that no one person can dictate where the bazaar will be – the group decides. True a king can say that everyone will do commerce only in one area, and network affects may then take over so that that lucky spot does become the central hub, but if the king chooses poorly, then that bazaar will eventually die in the face of market forces. To create a thriving bazaar, the first trick is to spot the place most natural for one to be.

Secondly, once one has convinced enough people that the bazaar is worth coming too, the bazaar will grow of its own accord. One will not need to convince people to come, since they will come out of their own best interest. Starting one may be difficult, but once one has gotten the process going, one can step completely out of the picture and the reaction will continue. All one has to do is play the role of a catalyst. Catalyzing the right solution, where the bazaar will grow, is the most important step.

CATALYZING A SOLUTION

WHAT DOES 'CATALYZING A SOLUTION' MEAN?

Linus Torvalds had one great advantage in creating Linux. He had no idea how big the thing would be when it was done. If someone said to him, "You are going to initiate the creation of a product that has millions and millions of lines of code, that people have invested literally millions of man hours working on," his response probably would have been "You're crazy." But this is exactly what has happened.

Below is his original invite to the community asking for help creating Linux.

> "I'm working on a free version of a Minix look-alike for AT-386 computers. It has finally reached the stage where it's even usable (though it may not be, depending on what you want), and I am willing to put out the sources for wider distribution… This is a program for hackers by a hacker. I've enjoyed doing it, and somebody might enjoy looking at it and even modifying it for their own needs. It is still small enough to understand, use and modify, and I'm looking forward to any comments you might have. I'm also interested in hearing from anybody who has written any of the utilities/library functions for minix. If your efforts are freely distributable (under copyright or even public domain) I'd like to hear from you so I can add them to the system."

So catalyzing the solution only requires that a core to build around has been created. In fact, creating too much may be counter productive. Programmers frequently join open source projects to help 'build the house' not to just put the shingles on the house that someone else built. What is important is that at every step along the way that the potential users see this software as 1.) Ownerless, and 2.) Useful.

In many ways people like Linus Torvalds and Eric Steven Raymond have shown us the way. The methods they have used to create successful open source projects and now completely open to see. If one has the wisdom to pick the right place for the bazaar, and the humility to study and follow in what they did, then the chance of success can only be greatly increased.

CONCLUSION

Hopefully this paper has made that case that it is highly desirable to have open source simulation software that helps the US Government agencies plan for coordinated action during complex operations. The author of this paper truly believes that the attraction of such software will be so great that it will eventually come to be. Both simulation and open source software are trends that are only accelerating in pace. So the convergence of these worlds colliding appears to be 'inevitable.'

But the inevitable can take a very long time. The sooner that we can bring these trend together to create a shared solution to aid in interagency coordination, the better. While it is hard to put a value on 'better coordination', 'better training' and 'better decisions,' it often becomes apparent after the fact how just a slightly different approach at key decision points can have profound implications down the line.

Someone has to set the stake in the ground and convince enough people that this bazaar is worth coming to. Hopefully this paper has done that. We will be working hard to create this bazaar. The value to people waiting in crises situations for the agencies to 'get coordinated' and get them the food, water, medicine or whatever else they need is just too great. Even if creating this bazaar saves just a handful of lives it will have been worth the effort. It will probably save many more.

REFERENCES

[1] http://www.state.gov/r/us/2007/88630.htm.

[2] http://www.au.af.mil/au/awc/awcgate/ndu/interagency_mgt_crisis_ops_2003.pdf.

[3] "Computer-simulated training on the rise" http://www.eschoolnews.com/news/showStory.cfm?Article ID=6742.

[4] http://sl.nmc.org/wp-content/uploads/2006/10/macarthur_press_release.pdf.

[5] Comment made at the Cornwallis XII meeting.

[6] The difficulty in creating a standard for sociological phenomena was pointed out by a participant at the Cornwallis XII meeting. It appears intuitively to be true.

[7] Quote extracted from the book "Open Source: A Mutlidisciplinary Approach" page 49.

[8] "Navy CIO OKs open source systems" http://www.gcn.com/online/vol1_no1/44441-1.html.

[9] "The 22 Immutable Laws of Marketing: Violate Them at Your Own Risk!" by Al Ries and Jack Trout.

[10] Raymond, E. (2001). The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolution (Rev. ed.). Beijing: O'Reilly & Associates Inc. Also available online at http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/.