



JAD on a Shoestring Budget

Dr. Mario J. Spina
The George Washington University

John A. Rolando
Science Applications International Corporation

Why is it seemingly so difficult to adequately address that first and all-important development phase – gathering and defining software requirements? The use of Joint Application Development offers more than a worthwhile and proven approach; it can be adapted to accommodate the business and administrative challenges in requirements gathering when there is seldom enough time and resources available to do it right the first time. The following article discusses the details of how the Science Applications International Corporation in McLean, Va. was able to inexpensively bring together users and developers to define complex and disparate requirements in a disciplined and effective manner. This laid a foundation for successful integration of application needs with existing commercial off-the-shelf and government off-the-shelf software tools and products.

The complexities of software development provide fertile ground for debate regarding which activities constitute its most critical steps and processes. However, many discussions on this subject suggest that the final frontier for successful software development is now, and may continue to be, the requirements gathering process.

For better or worse, gathering requirements demands involving the software application users, many of whom are neither educated nor experienced in the software development enterprise. Many software customers mistakenly believe that the up-front time spent in requirements gathering and analysis simply translates to an equivalent time delay in product delivery. This belief holds irrespective of studies and evidence showing that costs associated with correcting errors traceable back to poor up-front requirements, after fielding, can range from 68 to 200 times higher than the preventive costs associated with catching the errors during the requirements analysis phase [1].

With so much at stake, it is still surprising to learn how divergent the methodologies presented in textbooks and software journals are when discussing ways to elicit and rationalize software requirements. Few software engineering texts seem to provide much detail on how to elicit requirements, or otherwise may do so in one or more brief chapters or paragraphs offering only a short list of steps to consider in the process. Although referred to by a variety of names, a software requirements elicitation process called Joint Application Development (JAD) recognizes that requirements gathering is a very social endeavor [2]. It proposes a disciplined process for collecting, understanding, and organizing the innumerable and enigmat-

ic user perspectives inherent in a clear formulation and conceptualization of software needs.

Traditional systems design can result in disparate and conflicting requirements subject to mixed interpretations, and derived from the frequently limited participation of subject matter experts (who might also be the users). This precedes lengthy attempts to get feedback needed

“Costs associated with correcting errors traceable back to poor up-front requirements, after fielding, can range from 68 to 200 times higher than the preventative costs ...”

to reconcile and clarify the inevitable inconsistencies. As this article will attempt to show, the JAD approach is a proven course of action that may be ideally suited toward generating and organizing more clear and complete requirements from multiple users, when compared with traditional means of extraction [3].

Short History and Characteristics

JAD's lineage traces back to business systems' planning methodology developed by IBM in the mid-1970s. In addition to defining software requirements, this methodology also has been successfully applied to business planning, manufacturing, strategic planning, cost estimating,

test planning, and other domains. While variations to its use have evolved, some developers and software experts still consider it to be the best method for collecting requirements from the perspective of the users, customers, or customer advocates – classifying JAD as best practice for software projects [4].

In brief, JAD is essentially a structured workshop approach calling for a detailed agenda, a facilitator, comprehensive visual aids, and detailed record keeping. The characteristics of JAD, and keys to its success, are having committed participants, group cohesion, and the organization and structured setting just described.

Implementing a JAD session involves five phases. First, the project must be clearly defined in terms of purpose, functions, the team participants, and schedule. Second, some background research must be completed on user requirements and any anticipated problems and unique processes that might be required. Third, and particularly important, comprehensive visual aids should be utilized to help all participants better understand and visualize needs. Fourth, the session itself must be guided, all issues resolved, and all agreements well documented. Fifth, the session should be written up in a formal after-action report to be the basis for the software requirements [5].

Significant emphasis must be placed on two factors in this process. First, time, thought, and resources should be invested in having the best visualization tools as possible for the workshop [6]. Second, the commitment of top management is critical, especially in terms of the quality of, and direction given to, the session participants [7].

The Wireless Mobile Worker
Science Applications International Cor-

poration (SAIC) recently implemented the JAD concept in a unique way. The project is a valuable industry case study because it was successfully performed with very little budget. As a systems integrator, SAIC is not traditionally in the business of wholly developing commercial software products. This particular project focused upon the wireless mobile-worker market for online, automated procedures to perform test and inspection in the nuclear power industry.

Wireless mobile-worker technology in 1999 had an almost negligible installed base; capturing the customers' needs was critical to its acceptance in the market. To accurately do this, SAIC had to solve three problems. First, it had to have a rigorous forum from which to explain the concept and query its customers. Second, it had to get the customers to dialogue collectively with their engineers. Third, the SAIC engineering team had to find the funds required to support a conference for a thorough customer inquiry.

All of these problems were answered by one simple business development concept: teaming with industry. SAIC identified the key technologies and vendors providing solutions for elements of the procedure's automation business process. The areas of critical concern for SAIC's nuclear power plant customers were online interactive procedure development, document configuration management, real-time mobile computing, and data recording and reporting. No single vendor or product was able to offer the full life cycle solution.

SAIC reasoned that integrating existing products rather than attempting to reinvent the wheel could drastically reduce the costs of a full-scale software product development. Unfortunately, bringing the various vendors and products together in a full-scale integration was considered unattainable without some guaranteed number of paying customers. How better to guarantee customer involvement than to include them in the design and get up-front commitments? To help shepherd both the product vendors and power plant customers together, SAIC evolved the concept of a JAD-based conference called the Procedure Automation Consortium.

For the conference, vendors were to supply their products and the necessary application program interface code, and SAIC was to perform the role of technology solutions architect and systems integrator. To communicate the idea and kick-off the conference, SAIC hosted the JAD meeting in cooperation with a

nationally recognized nuclear utility services provider. The three-day event, held in May of 1999 at the upscale Arlington Hilton Hotel, brought 12 product vendors together with 16 power plants for a total participation of approximately 80 industry professionals, split evenly between customers and vendors.

JAD in Action

Each day of the JAD conference began with a keynote talk by a prominent industry figure having a strong grasp of technology trends in the field. The first two mornings, attendees viewed product presentations from each vendor in four separate conference rooms, one for each of the business process specialties: procedure development, document management, mobile procedure execution, and data recording and reporting. The first two afternoons attendees were asked to indicate what they liked and did not like about the products they had seen during the morning sessions.

Professional facilitators worked to cultivate customer and vendor interchange and turn customer comments into

“Professional facilitators worked to cultivate customer and vendor interchange and turn customer comments into system requirements.”

system requirements. The facilitators were supported by stenographers and employed an on-screen requirements database. The full-screen view of each requirement, displayed as simple, stand-alone *shall* statements allowed customer groups an opportunity to specify, negotiate, and validate the precise phrasing of each system requirement. The facilitators focused upon building requirements consensus through technical discussion and point negotiations.

On the third and final day, the results of the previous two days were consolidated and presented in summation to the collective audience. Before lunch, customers were asked to rate the products they had seen, identify a value and a desire to purchase such products, and indicate their willingness to fund a political action committee effort to develop a product that would achieve the set of

requirements identified at the conference. As a structured workshop approach, JAD is normally performed in relatively small classroom-sized settings. Having 80 participants placed a premium on visual aids, experienced facilitation, and opinion management. The large full-screen display was critical.

Consortium Results

The full affair was well done with three quality meals each day and full-time coffee, tea, and soda service. The total bill for the event was approximately \$35,000, but SAIC paid only a small portion of the costs. The product vendors were willing to sponsor meals and advertisement banners, and attendees were willing to pay a small \$100 conference fee. Overall, the venture was well worth the effort, not only for its informational value but also for the industry goodwill it engendered and the networking opportunities it offered.

The data from the survey was extensive. Customers had not only indicated their requirements and the products they preferred, but also indicated an initial willingness to purchase the associated software solution. SAIC was thereby armed with the information needed to select integration partners, the product requirements, the purchase price, and the business probability data from which to make a project go/no-go decision.

With this information, a COTS application architecture was defined; key vendors provided cost estimates for the licensing, interface development, and integration of their products into a final solution. With an assumption that initial development and roll-out costs could be spread across the set of customers who indicated a high likelihood of product purchase, a per customer solution cost was estimated. From this estimate, the SAIC architect was quickly able to determine that, unfortunately, product costs significantly exceeded customer perceived value. Accordingly, this product development effort was cancelled.

While this JAD did not result in a successful software product development, it presented some valuable information and experience. The effort showed how the JAD approach could be used when there is a need to develop and refine a software solution that integrates an existing product base. It also showed how a software requirements elicitation involving software users, product vendors, and a systems integrator could be accomplished on a shoestring budget. Lastly, there was the advantage of having multiple poten-

tial customers together to analyze and assess the demand and economic viability of further development.

Epilogue

JAD is, of course, one of a multitude of requirements elicitation techniques. We have found no actual software developmental data comparing the use of JAD with alternative procedures and techniques in terms of relative successes or failures. In addition, since all projects are somewhat unique, we believe any comparative empirical data would somehow have to be adjusted for the multitude of other diverse variables to be meaningful.

It would seem intuitive that there would be advantages in having a dedicated gathering of users, developers, and customers together in a structured setting, compared with shorter piecemeal sessions or multiple one-on-one sessions. However, it is also easy to imagine circumstances whereby just the converse would be true, i.e., that shorter piecemeal sessions or multiple one-on-one sessions would be better if, for example, the JAD facilitator was somehow skewing inputs directly or indirectly via the recording process [8]. Thus, before any reader commits to using the JAD approach, we suggest they perform their own analysis and thought. We offer the following list of sources and Internet sites for further research. ♦

References

1. Weigers, Karl E. Software Requirements. Redmond, Wa.: Microsoft Press, 1999: 15.
2. Wood, Jane, and Denise Silver. Joint Application Development. New York: John Wiley & Sons, Inc. 2nd ed., 1995: 30-31. Here are some synonymous examples: facilitated meeting, facilitated session, facilitated design, joint application design, joint application development, joint application review, joint application planning, et al.
3. Ibid. 5-6.
4. Cline, Allan. "Joint Application Development (JAD) for Requirements Collection and Management." White Paper. Carolla Development. Available at <www.carolla.com/wp_jad.htm >.
5. Weigers, Karl E. Software Requirements. Redmond, Wa.: Microsoft Press, 1999: 8-9.
6. August, Judy H. Joint Application Design: The Group Session Approach to System Design. New Jersey: Yourdon Press, 1991: 5.
7. Wood, Jane, and Denise Silver.

Joint Application Development. New York: John Wiley & Sons, Inc. 2nd ed., 1995: 166.

8. Christel, Michael G., and Kyo C. Kang. "Issues in Requirements Elicitation." Technical Report CMU/SEI-92-TR-12. Sept. 1992, ESC-TR-92-012. Available at <www.sei.cmu.edu/publications/documents/92.reports/92.tr.012.html>.

Additional Reading

1. August, Judy. Joint Application Design. Englewood Cliffs, N.J.: Yourdon Press, 1991.
2. Garner, Rochelle. "Why JAD Goes Bad." Computerworld 10 April 1995.
3. Hollander, Nathan, and Naomi Mirlocca. "Facilitated Workshops: Empowering the User to Develop Quality Systems Faster." Industrial Engineering Oct. 1993.
4. Knowles, Anne. "Peace Talks: Joint Application Development." PC Week 11 Dec. 1995.
5. Leventhal, Naomi. "Using Groupware Tools to Automate Joint Application Development." Journal of System Management Sept.-Oct. 1995.
6. Martin, James. Rapid Application Development. New York: MacMillan Publishing Company, 1991.
7. McConnell, Steve. Rapid Development. Redmond WA: Microsoft Press, 1996.
8. Wetherbe, James C. "Executive Information Requirements: Getting It Right." MIS Quarterly Mar. 1991.
9. Wood, Jane and Denise Silver. Joint Application Development. New York: John Wiley & Sons, Inc. 2nd ed., 1995.

Web Sites

- <www.utexas.edu/hr/is/pubs/jad.html>.
- <www.dci.com/events/jad>.
- <www.credata.com/research/jad.html>.
- <www.russellmartin.com/courses/JAD.htm>.
- <www.carolla.com/wp-jad.htm>.
- <csweb.cs.bgsu.edu/maner/domains/RAD.htm>.
- <www.trainersdirect.com/outlines/JAD.htm>.
- <www.sisg.com/JAD.htm>.
- <www.verhoef.com/cs/njrad.htm>.
- <www.computer.muni.cz/cspress/CATALOG/rs00072.htm>.
- <www.mgrossmanlaw.com/articles/mhlt/joint_application_development.htm>.

About the Authors



Mario J. Spina, D.Sc., is a part-time faculty for The George Washington University School of Business and Public Administration and the School of Engineering and Applied Science, as well as deputy division manager for Science Applications International Corporation's (SAIC) Aerospace Technology Applications Division. As a new product development manager for SAIC, Dr. Spina led the company's charge into the wireless mobile worker software market. He has a doctorate of science degree in engineering management from The George Washington University, a master's of science degree in electrical engineering from the University of Southern California, and a bachelor's of science degree in mechanical engineering from California State University, Northridge.

**310 Cabin Rd. SE
Vienna, VA 22180
Phone/Fax: (703) 242-6732
E-mail: mspina@gwu.edu**



John A. Rolando is currently a senior operations research analyst for Science Applications International Corporation (SAIC) Information Technology Solutions Group.

He has been directly supporting communications interoperability analyses and the development of two detailed communications models for the Command, Control, Communications, and Computer Systems Directorate, Joint Staff, the Pentagon, since 1996. He has a bachelor's degree in economics from LeMoyne College, Syracuse, a master's degree in public administration from the University of Northern Colorado, and a master's degree in business administration from Marymount University, Arlington, Va.

**Science Applications International Corporation
1710 SAIC Drive
McLean, VA 22102
Phone: (703) 676-4682
Fax: (703) 356-2534
E-mail: john.a.rolando@saic.com**