ANL/DIS/CP- 95613

AN EVALUATION OF SMALLTALK, C++, AND ADA95 FOR USE IN A MAJOR DoD APPLICATION

John R. Hummel and Lucian Russell
Decision and Information Sciences Division
Argonne National Laboratory
9700 S. Cass Avenue/Bldg-900
Argonne, IL 60439-4832
e-mail: hummelj@smtplink.dis.anl.gov

KEYWORDS Smalltalk, C++, Ada95, Programming Languages, Object-oriented

ABSTRACT

The choice of a programming language can have a significant impact on the development of effective software. Numerous programming languages are available in the computer science community and supporters of specific languages often defend their choices with a religious zeal.

Argonne National Laboratory (ANL) was recently tasked to assist in the selection of a programming language and development environment for a major DoD software development effort. Three candidate programming languages were identified by the sponsor for evaluation: Smalltalk, C++, and Ada95. (At the time the evaluation was performed, Java was regarded as being too immature.) Smalltalk is a pure object-oriented language that was originally developed by Xerox. C++ originated at AT&T and is not a pure object-oriented language. Ada95 is an object-oriented version of Ada, a language that was developed to support the defense community. The purpose of this paper is to describe the process that was used to evaluate the three languages and to describe the results of the evaluation.

INTRODUCTION

The evaluation of the programming languages and development environments was performed as a three-part effort. The first part of the assessment determined how the selection of a programming language would affect the program over the life of the program. The second part examined the experiences of the teams that developed prototype systems for the sponsor with regard to the languages they used in their efforts. The third part involved the collection and evaluation of data and information from the Modeling and Simulation (M&S) community — both DoD and civilian — on the usage, performance, cost, and quality of software developed in the candidate languages. This last part of the evaluation was performed by reviewing available trade and refereed journals, and Internet sites, and by interviewing users. The information collected was a combination of quantitative and qualitative findings.

IMPACT OF LANGUAGE SELECTION ON PROGRAM GOALS

A programming language and its development environment represent the tools used to implement the software design quickly and efficiently, provide software maintenance quickly and efficiently, detect and correct defects in the software (*i.e.*, reduce the mean time to repair), and assist in meeting the performance goals of the program. In the case of the program that requested the language evaluation, the choice of a programming language impacted other decisions, such as the selection of database management systems as well as any commercial packages. Finally, the choice of programming language impacted the amount of the prototype software that could potentially be reused in the production version. Therefore, the selection of a programming language was based on a business-case evaluation of which language would provide the best value over all phases of the program: software development, routine use, and maintenance.

Implementing the Software Design

The software being developed for the program is intended to be an object-oriented (OO) simulation system. Table 1, which was obtained from multiple sources, including the Ada Information Clearinghouse World Wide Web site and the report "Smalltalk Market Accelerates" by the International Data Corporation (McClure 1995), compares the object-oriented features of the three candidate languages. The conclusion to draw from this Table is that any of the three candidate languages could suffice although the "purity" of the object-oriented design varies from one language to another.

In implementing the design for a software system, there is no guarantee that a single programming language can provide optimal features in all areas. The use of a mixed language environment can provide a better way to meet all program goals by using the better language where appropriate. The use of a mixed language environment can also decrease program costs and increase reuse potential by enabling legacy or other reusable software to be integrated "as is" in the language that it was developed in.

Table 1. A comparison of the object-oriented features of Ada95, C++, and Smalltalk.

FEATURE	ADA95	C++	SMALLTALK
OO Language Type	Pure	Hybrid	Pure
Strong Typing	Y	Y*	N
Compile Time Checking of Errors	Y	Y	Y
Single Inheritance	Y	Y	Y
Multiple Inheritance	Y**	Y	Y***
Polymorphism	Y	Y	Y
Namespace Management	Y	Y	N
Exception Handling	Y	Y	Y
Hierarchical Libraries of Namespace	Y	Y	N
Concurrency	Y	3rd Party	Y
Distributed Programming	Y	3rd Party	3rd Party
Parameterized Components	Y	Y	N

*Can be overridden by the user.

**Restricted to three common uses of multiple inheritance that are supported in Ada95 through a combination of existing Ada83 and new Ada95 facilities.

***Available through a variant of Smalltalk known as ClassTalk

EXPERIENCES FROM THE M&S COMMUNITY

Overview of the Languages

C++ originated at AT&T Bell Laboratories in the mid-1980s. It is a compiled language, and in 1990 it was estimated to have held 90% of the object-oriented programming market (Radding 1994). It is statically typed and an object-oriented extension of the C language. Different compilers and tools are available from a variety of vendors and third party providers, including AT&T, Sun, Microsoft, Borland, Centerline, IBM, and GNU, among others.

Smalltalk originated at the Xerox Palo Alto Research Center in the mid 1970s. Smalltalk was originally intended to be a programming language for children and had its roots in Simula 67, a language developed for modeling and simulation. Smalltalk is a simple, dynamically typed, pure object-oriented programming language that is interpretive (*e.g.*, "just in time" compiling). Compiled versions of software can be produced after development has been completed. The use of Smalltalk has been growing rapidly, with a 60% growth in 1994 relative to 1993 (Shan 1995). Compilers are available from a variety of vendors, including ObjectShare, Inc. (formerly ParcPlace/Digitalk), IBM, and GNU.

Ada, developed in the late 1970's to provide a standard language for DoD applications, is a compiled language. The latest version is Ada95, which is object-oriented. Ada has limited use in the commercial sector and is mandated for use in specific DoD applications, such as embedded warfighting systems.

All three of the languages have undergone some degree of standardization. Also, each language is available on a wide range of platforms.

Programming Language Use in the M&S Community

Table 2 lists the programming languages used by a sampling of DoD simulation systems. In addition to the applications noted in Tables 2, two other DoD programs were identified as using Smalltalk. The first, at the National Security Agency, involved an application operating in the artificial intelligence area. The second, at the Naval Research Laboratory, involved the managing of the Navy's first object-oriented digital mapping project (Shaw et. al. 1996). In other parts of the U.S. government, the Social Security Administration Agency has purchased a large number of Smalltalk development licenses.

Outside of the DoD community, C++ is widely used, with Smalltalk use growing rapidly. There is greater use of Smalltalk in the civilian sector than in the DoD sector. Organizations using Smalltalk extensively include Electronic Data Systems, Anderson Consulting, American Airlines, American Management Systems, Xerox, Allen-Bradley, Bell South, Norfolk Southern Railway Company, Florida Power and Light, FedEx, Bell Atlantic, and Credit Suisse, to name just a few.

Table 2. Programming languages used by a sampling of DoD M&S applications.

SIMULATION SYSTEM	PROGRAMMING LANGUAGE
Dynamic Information Architecture System (DIAS)	Core Architecture in Smalltalk, Augmented by C Extensions to C++. Integrated Physics Models or Applications are Kept in Their Native Languages.
Joint Warfare System (JWARS)	Smalltalk
Eagle	Lisp
Naval Simulation System (NSS)	C, C++, and Assembler.
J-MASS	C++ (Release 3.0)
ARES	C++
ModSAF	C
JointSim	C++
FLAMES	C, C++

Florida Power and Light (FPL) reported having from 30 to 40 applications written in Smalltalk. All are deemed critical applications, and the number of users using the applications range from a few to more than a thousand users. FPL uses C++ for small applications but not for large ones. FPL also operates in a mixed language environment, with Smalltalk as the base language. FPL noted that if significant floating point operations were encountered in an application, FORTRAN was preferred over C++.

FedEx uses Smalltalk in a variety of applications. One is a GIS application used in international route planning and optimization. Their dispatching system for scheduling pickups and deliveries (based on scheduling algorithms and heuristic balancing) is also in Smalltalk. Finally, a new system has been developed that will be able to do "on-the-fly" analysis of the status of packages enroute, including the identification of packages that are at risk of not being delivered on time. FedEx reported that they operate in a mixed language, with Smalltalk as their base language.

In a study for the Smalltalk Industry Council, International Data Corporation (IDC) did a survey of how C++ and Smalltalk are used in software applications. Table 3, which lists the results of the IDC study, shows that both languages are used in a variety of different applications.

Programmers' Perceptions of the Languages

Each programming language has a camp of devoted followers and finding an objective technical evaluation of the languages was difficult. Many evaluations are either produced by a vendor or by an individual or group with an obvious bias one way or another. Table 4 summarizes the *perceived* strengths and weaknesses of the languages evaluated.

The IDC also surveyed 300 programmers from four software communities (Smalltalk, C++, COBOL, and 4th Generation Languages (GL)) of their perceptions about Smalltalk and other languages and supporting products. The results of that study, performed in 1994, are shown in Table 5. The programmers rated each topic on a scale of 1 - 5, with 1 as a major competitive disadvantage and 5 as a major competitive advantage. The language with the highest rating in each category is listed; an asterisk denotes a rating of 4 or higher. In this survey, Smalltalk was rated highest in a given category more often than were the other languages considered.

Ease in Learning the Software

Smalltalk is often reported as being easier to learn than C++. Data collected for the Smalltalk Industry Council bear this out. These claims are also supported by the empirical experiences of Argonne personnel in relation to the Dynamic Environmental Effects Model development effort after ANL made the switch from C++ to Smalltalk. Three senior-level ANL programmers attended a one-week Smalltalk training course and then retrained the remaining members of the development team in about two weeks.

The Selection Process for Languages

IDC also surveyed groups using C++ and Smalltalk and asked them to state if the language was selected through management edict, a formal review process, or an informal process. Smalltalk was selected following a formal review process in 51.9% of the cases, while for C++ the corresponding value was 21.3%. C++ was selected informally in 62.7% of the time, while the corresponding value for Smalltalk was 31.4 %. The IDC report concludes "...when people really study the issue, they choose Smalltalk."

Table 3. Application mixes, in %, for C++ and Smalltalk usage.*

APPLICATION	C++	SMALLTALK
Real-time/Process Control	12.4	8.0
Office Automation/Personal Productivity/ Groupware	13.5	12.0
Primarily MIS/Transaction Processing	18.5	19.2
Information Retrieval/Reporting/Query/ Decision Support	31.9	54.3
Scientific/Engineering/M&S	23.7	6.5

*The study involved a survey taken in 1994 of 92 users of C++ and 56 users of Smalltalk.

Table 4. Perceived strengths and weaknesses of C++, Smalltalk, and Ada95.

PERCEIVED STRENGTHS	PERCEIVED WEAKNESSES
C++	**C++**
Better Performance	Not Pure Object-Oriented
Perceived as Standard	Difficult to Learn
Platform Integration	Evolving Nature of Language
Cross-Platform Development	Few Development Tools and Poor Development Environments
Allows Access to Low-Level System Features	Lack of Standard Libraries
Perceived to be Similar to C	Compiler Differences
Smalltalk	**Smalltalk**
Development Productivity	Environment and Applications are Integrated
Environment and Applications are Integrated	Arithmetic Performance
Easy to Learn	Limited Third-Party Libraries
Enforces Object Paradigm	Used Primarily for MIS Applications
More Dynamic	
Integrated Class Libraries	
Platform Independence and Portability	
Ada95	**Ada95**
Best Software for High-Assurance, Real-Time Applications	Insufficient Track Record to Gauge True Abilities
Some Object-Oriented Support	Weak DoD Support for Infrastructure Development
	Weak Acceptance by Commercial Sector

PRODUCTIVITY IMPACTS

One of the reasons often mentioned for selecting Smalltalk over C++ is the productivity improvement afforded by Smalltalk's interpretive nature. It is significant to note that, in the IDC study, Smalltalk was rated highest by the programmers surveyed in the areas of enforcement of the object-oriented paradigm, reliability of deployed applications, rapid application development, and cost-effective deployment. For example, John Radford of DNA Enterprises, Inc., reported at OOPSAL '95 that they found Smalltalk to be 2-3 times more productive, as measured in terms of final cost, than C++. He also mentioned that when they bid on contracts involving C++, the contracts are generally bid at twice the cost of those based on Smalltalk. Also, Electronic Data Systems replicated in Smalltalk an existing manufacturing application originally written in PL/1. They experienced an increase of 3 to 4 times in productivity related to the design and implementation of the software. They also observed little or no decrease in performance in the final system. Finally, the Canadian Defense Research Establishment reported a 6-to-1 advantage in using Smalltalk.

The degree of productivity increase experienced was variable, and often it was not clear what programming language Smalltalk was being measured against. FPL reported that they do not develop many applications in C++ because those projects do not deliver on time, whereas Smalltalk projects do. Another developer, at the National Security Agency, reported that the rapid development environment of Smalltalk was essential to their application involving employing artificial intelligence. The real-time responsiveness required by their application, such as retraining neural nets as new data come in, could not be provided by C++.

PERFORMANCE COMPARISONS

Performance of a software system is a function of four factors: the context of the problem being considered, the design of the software, the hardware platform being used to perform the calculations, and the programming language being used. Measuring the performance of an application is important, but it must be done carefully because designing the software well is as important as using the best performing language (*i.e.,* poorly designed software can be used with a highly efficient language.)

The Smalltalk users contacted for this study did not express any concern over the performance of Smalltalk in their applications. If a performance problem was found it was usually traced to a problem in the design of the application or an improper use of Smalltalk. If the problem could not be fixed by a design change, a mixed language solution was typically used. FPL reported that if arithmetic performance problems were encountered, they would select a FORTRAN solution before they would turn to C++.

SUMMARY

Table 6 gives an overall summary of the three programming languages evaluated. The features are rated as positive (+) or negative (-), and the positive features are shaded for ease of review. Smalltalk and Ada95 provide a "purer" OO environment for software development than does C++. This means that an application can be designed and then implemented in a pure OO context.

Smalltalk, being an interpretive language, offers more rapid development, although the degree of productivity improvement is highly dependent on the application and the skill of the developers. In addition, Smalltalk offers a fully integrated development environment, meaning that one need only acquire one package rather than having to buy several, as is the case with C++. Because of the newness of Ada95, the development environments available seem to be more limited.

There is good object database support for C++ and Smalltalk but little to none for Ada. There is also good vendor support for C++ and Smalltalk and less for Ada95. More third party tools are available for C++ than for Smalltalk, but many of them are required because of the nonintegrated nature of the C++ development environment. Also, some of the C++ tools are required to perform functions to correct memory leakage inherent to C++ but not experienced in Smalltalk.

Both C++ and Smalltalk offer good cross-platform use. The degree of cross platform compatibility for Ada95 is not known.

The arithmetic performance of C++ is better than that of Smalltalk although the two major Smalltalk vendors are making improvements in this area. No data could be obtained related to arithmetic performance of Ada95.

The prototype system that was developed for the program sponsor was based on Smalltalk in a mixed language environment, a common practice in the M&S community. It was recommended that the production version of the application continue to be developed with Smalltalk in a mixed language environment. This kind of approach is consistent with development practices described by different users. In addition, the use of a mixed language environment maximizes productivity and performance by enabling the best features of different programming languages to be used when and where appropriate

REFERENCES

McClure, S. (1995) "Smalltalk Market Accelerates," International Data Corporation, IDC #9819, December.

Radding, Alan (1994) *Computer World,* May.

Shan, Yen-Ping (1995) "Smalltalk on the Rise," *Communications of the ACM*, Vol. 38, No. 10, pp. 103-104, October.

Shaw, Kevin, Cobb, Maria, Chung, Miyi, and Arctur, David (1996) "Managing the US Navy's First OO Digital Mapping Project." *IEEE Computer*, pp. 69-74, September.

Table 5. Survey results from 300 programmers of Smalltalk, C++, COBOL, or 4th Generation Languages who evaluated the languages on a 1 - 5 scale. The highest rating in each category is noted. (asterisk denotes a rating of 4 or higher.)

Evaluation Topic	**Smalltalk**	**C++**	**COBOL**	**4GL**
Support for OO Development	X*			
Enforcement of the OO Paradigm	X*			
Cost of Run-time Licenses	X	X		
Integration of Development Environment	X	X		
Development of Scientific, Engineering, Modeling and Simulation Applications	X	X		
Portability to Many Operating Systems	X	X		
Development of Real-time, Process Control applications	X	X		
Degree of Proprietary Versus Open	X	X		
Reliability of Deployed Applications	X		X	
Rapid Application Development	X			X
Cost-effective Development	X			X
Development of Client/Server, Information-Retrieval, Query, Decision-Support Applications	X			X
Development of Office Automation, Personal Productivity, Groupware Applications	X	X		X
Ease of Learning Syntax	X		X	X
Overall Simplicity of Language	X		X	X
Ease of Code Maintenance	X		X	X
Support for Development of Large Complex Applications	X	X	X	X
Availability of Mature Class Libraries	X	X	X	X
Support for Multi-Developer Teams	X	X	X	X
Performance of Runtime Applications		X*		
Compatibility with Other Packages/Software		X	X	
Integration with Legacy Applications and Databases			X*	
Maturity of the Language			X	
Maturity of the Available Development Tools			X*	
Availability of Trained Developers for Hire			X*	
Product Documentation and Vendors Support			X*	
Development of Primarily Transaction Processing Applications			X	
Number of Times Rated Top in a Category	**19**	**12**	**14**	**10**

Table 6. Summary comparison of capabilities of C++, Smalltalk, and Ada95.

CATEGORY	C++	Smalltalk	Ada95
Support of Object-Oriented Design	-	▨	▨
Rapid Development Environment	*	▨	-
Full Development Environment (*i.e.*, Do Not Need Third Party Tools)	-	▨	?
Supports Object Database Management Systems	▨	▨	-
Broad Vendor Support	▨	▨	-
Support for Efficient Configuration Management (*e.g.*, ENVY)	▨	▨	-
Supports Cross-Platform Use	▨	▨	?
Good Arithmetic Performance	▨	-	?
Good Nonarithmetic Performance	▨	▨	?

*Smalltalk-like development environments for C++ are being developed.