LBNL-51548
IO-461

Presented at the European Community Product and Process Modeling Conference (ECPPM 2003) on September 11-13, 2002 in Portoroz, Slovenia.

EARLY LESSONS FROM DEPLOYMENT OF IFC COMPATIBLE SOFTWARE

V. Bazjanac

Building Technologies Department
Environmental Energy Technologies Division
Ernest Orlando Lawrence Berkeley National Laboratory
University of California
1 Cyclotron Road
Berkeley, California 94720-8134 USA

May 1, 2002

This work was supported in part by the Assistant Secretary for Energy Efficiency and Renewable Energy, Office of Building Technology, Building Technologies Program, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098, and internal funding from Lawrence Berkeley National Laboratory

EARLY LESSONS FROM DEPLOYMENT OF IFC COMPATIBLE SOFTWARE

V. Bazjanac

Building Technologies Department, Environmental Energy Technologies Division
Ernest Orlando Lawrence Berkeley National Laboratory

ABSTRACT

The Industry Foundation Classes (IFC) model of the International Alliance for Interoperability (IAI)—an object data model of buildings—is in its seventh year of development. The last three releases of the model (IFC 1.5.1, 2.0 and 2x) have been implemented by a number of "mission critical" industry applications. The deployment of such software in real life projects is just starting. The author is exploring lessons from early deployment that are related to end user and general industry readiness for software interoperability, project model population with data and issues with compatibility of project data, built-in limitations in applications and in the data model, exchange file size and the selection of interoperable software for a project, as well as benefits attainable today from the use of interoperable software. He concludes that software interoperability is beginning to work in this industry, although not as smoothly as first expected.

INTRODUCTION

The International Alliance for Interoperability (IAI) has been developing its Industry Foundation Classes (IFC), a universal object-oriented data model of buildings, for almost seven years. What started as a relatively small and exclusive architecture, engineering, construction, and facilities-management (AEC/FM) industry consortium, the *Industry* Alliance for Interoperability, in North America in 1994, modified its name and became an open and international not-for-profit industry alliance a year later (in 1995) that now includes about 600 members worldwide (IAI 1995a). The membership includes organizations and individuals that are architects, engineers, contractors, building owners, facility managers, material and equipment manufacturers, software vendors, information providers, government agencies, research laboratories, universities and more.

The mission of the IAI is to "…define, publish and promote specifications for Industry Foundation Classes (IFC) as a basis for project information sharing in the building industry (architecture, engineering, construction, and facilities-management). The information sharing is worldwide, throughout the project life cycle, and across all disciplines and technical applications." (IAI 1995b) In practical terms this has a dual intent: (a) to facilitate information exchange in the AEC/FM industry by enabling industry software interoperability, and (b) to create a *de facto* standard for the AEC/FM industry that defines how information is formulated and exchanged electronically.

The meaning of interoperability for the IAI is *software* interoperability. Three ingredients are necessary for any software interoperability to function: (1) a data model, (2) software implementations of the data model, and (3) deployment of that software (**Fig. 1**). The three ingredients are inseparable; they feed on each other and interoperability is not attainable if one is missing.

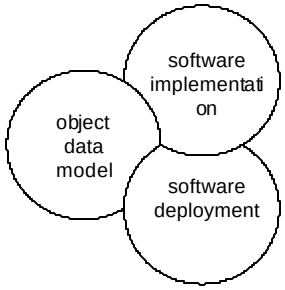

Figure 1. Ingredients necessary for software interoperability.

A data model developed today must be intelligent and extensible. This implies the use of object oriented technology, modularity and extensible model architecture. Software implementation in the AEC/FM industry is largely confined to the development of interfaces to the data model for *existing* software; development of brand new interoperable software is rare. An exception to this is software made possible by the existence of the data model, such as Solibri Model Checker (Solibri 2001). Software deployment is the *use* of software by industry professionals in performing their regular daily work and tasks on real life projects.

The IAI has spent most of its effort so far on the development of its object data model, IFC. This is a universal model of buildings that standardizes how buildings are defined electronically from every AEC/FM industry professional point of view. It is intelligent and extensible. While any large and complex data model is never complete and is always in the state of "work in progress," the latest IFC release (IFC2*x*) offers some very useful functionality. It is now possible to exchange building geometry of almost any arbitrary complexity, as well as some rudimentary information about project and facilities-management, HVAC and electrical equipment, furnishings, and more. Model functionality will expand significantly with the release of IFC2*x* Second Edition (IFC2*x*2), planned for the first quarter of 2003.

The last three releases of the data model (IFC 1.5.1, 2.0 and 2*x*) have been implemented in a variety of industry software over the last couple of years. This has been accomplished mostly by members of two groups: the IAI Implementation Support Group (IAI_ISG 2001) and the Building Lifecycle Interoperable Software group (BLIS 2000). The focus of the former is mostly data exchange among CAD applications; it is composed primarily of CAD vendors and their implementations are based on IFC releases 1.5.1 and 2*x*, with IFC2*x* based software finally coming to the market later this fall. The latter is mostly composed of "downstream" vendors with a focus on building lifecycle; these implementations are based on IFC 2.0. A number of software developers are members of both groups.

Most of software that implemented IFC to date is of the "mission critical" type: professional software that is critical to conducting business in a segment of industry. The IFC object data model is already too large for implementation in its entirety. IAI implementers have reached agreements on what information their software should be able to exchange and, consequently, what subset of the data model to implement in their individual software to enable the agreed-upon exchange of information. These agreements are expressed in implemented "views" of the model. Views supported to date reflect parts of the work flow in the industry: "CAD," "architectural design to quantity take-off and cost estimating," "HVAC design to quantity take-off and cost estimating," "architectural design to thermal load calculation and HVAC design," and "client brief and space layout to architectural design" (BLIS 2000; IAI_ISG 2001). Additional views will emerge in the future with the implementations of IFC release 2*x*2.

As an organization, the IAI has done very little to directly support software deployment in the industry to date. The IAI End User Support Group never gained momentum and was declared inactive by the end of 2000. As newly interoperable industry software is released, individuals and individual organizations within the IAI are embarking on "pilot projects." Such efforts are trying to use small groups of interoperable software to accomplish a variety of professional tasks on real-life projects.

DEPLOYMENT OF INTEROPERABLE SOFTWARE

While the number of implementations is still relatively small, a few real-life industry projects have lately employed such software with the expectation of benefiting from the promised direct data exchange. The IAI "pilot project" web site lists quite a few projects (IAI 2002). Only a handful of those represent true deployment of IFC compatible software in industry projects; most listed

projects are actually development projects of some kind.

True deployment of IFC compatible software involves the use of such software in real life professional work. Deployment projects exchange data about a real building (i.e., one that is in design or was built) among *several* applications. This means that the end users are using multiple software applications to accomplish, individually or as a group, a set of professional tasks they are expected to do, such as to design the building or building systems, create renderings and other visualization media to communicate something about the building, prepare a cost estimate, plan the construction sequence, operate the building or manage the project.

It is difficult to determine how many software deployment projects are in progress. Several projects are kept private for a variety of reasons, with very little or no information about them released. Only three deployment projects with information available to the public have been exposed to date.

Helsinki University of Technology—HUT 600

The famous auditorium designed by Alvar Aalto in 1960s needed enlargement to 600 seats - thus the project name "HUT 600" (**Fig. 2**). The multidisciplinary effort involved the use of several software applications in the design and planning of the reconstruction (Iso-Aho 2002a). The use of interoperable software was monitored (Kan, C., in prep.). The building model is based on IFC 2.0; IFC compatible software aided in visualization, lighting simulation, comfort analysis, simulation of energy performance, air flow analysis, and environmental and lifecycle cost analyses. In addition, other software was used for 4D construction planning and visualization.

Headquarters for the Danish Broadcasting Corporation

This is a design for a geometrically complex building of 50,000 m^2 (**Fig. 3**). The project goals, among other, were to demonstrate the use of IFC

Figure 2. Auditorium HUT 600 at the Helsinki University of Technology. Architect: A-KONSULTIT OY

compatible software with real project data, and to develop an end user guide for those who wanted to use such software in their projects (Karlshøj 2002). The three-dimensional building model was developed and used by nine applications (three of them CAD) for model viewing and checking, thermal performance analysis, quantity take-off and cost estimating.

Figure 3. Headquarters for the Danish Broadcasting Corporation

LBNL E-Lab Building

The preliminary schematic design for a new five-story 3000 m^2 laboratory building (**Fig. 4**) at the Lawrence Berkeley National Laboratory (LBNL) in Berkeley, CA includes unusual features, such as two-story rotating laboratories and segments of replaceable façade (Bazjanac 2002). The three-dimensional building model is based on IFC release 2.0; five IFC compatible software

Figure 4. Preliminary schematic design for the E-Lab building at the Lawrence Berkeley National Laboratory. Architect: Stanley Saitowitz.

applications used the model for model checking, visualization studies, energy performance analysis and cost estimating. Four other applications that can indirectly import building geometry through other IFC compatible software from which they can import data, were also deployed in work on additional tasks, such as lighting simulation and additional high end visualization. LBNL hired experts to operate software or extend data bases for some of the used applications.

Even though interoperable software deployment efforts are still in their infancy, a number of important lessons are already emerging. While the author is drawing his findings primarily from his involvement with and experience from the E-Lab project at LBNL, very similar findings have been reported in personal communications and oral presentations by individuals who directly participated in the other two deployment projects (Karlshøj 2001; Backas 2002; Iso-Aho 2002b).

LESSON 1: THE INDUSTRY AT LARGE IS STILL UNPREPARED

It is clear that the industry is still largely unprepared for software interoperability. The average end user does not "think" in terms of three-dimensional building *models* yet, and is lost when asked to define information required by and according to the rules of interoperable software. The dominating mind-frame in the AEC/FM industry is still two-dimensional; even architects, trained to visualize in 3-D, still "think" in 2-D (i.e., conceptualize and sketch in terms of 2-D plans, sections and elevations). Some CAD vendors' attempts to convince their end user community to switch to three-dimensional modeling have apparently had only limited success—while some vendors claim that sales of such software are substantial, there is no evidence that the actual creation and submission of three-dimensional project models has markedly increased. Anecdotes abound in the architectural design community where end users purchased new three-dimensional modeling CAD software only to use the built-in drawing engines to draw in 2-D.

Such end users face a *very* steep learning curve with typically little time available to learn. Practicing professionals, faced with day to day project deadlines, seldom have the time, resources or institutional support to retrain—they must learn "on the job." Learning modeling in 3-D and fundamentally changing how they do their daily work is too much for them to handle. It will probably first take a new generation of consultants to show industry the benefits of changing the work paradigm, and a new generation of educators to teach future professionals how to do it before three-dimensional modeling of buildings becomes widespread. Government demands on project submission formats may accelerate this process, as seems to be the case in Singapore (Mok 2002).

For the time being, success in deployment of interoperable software and gains in real life projects seem possible only if *teams of experts* do most of the work. These are individuals who are experts in *both* the profession they represent and in the corresponding professional software, working together (see LESSON 8 below). The number of such experts is still small and it may be difficult and costly to assemble such expert teams; this may be more than offset by the long range benefits of successful deployment of interoperable software (see LESSON 7 below).

LESSON 2: POPULATING THE PROJECT MODEL WITH DATA IS NOT ALWAYS EASY

Populating the project model properly with data can cause significant difficulty. Most downstream applications expect upstream definition of at least some of the data they use. Since the definition of building geometry is exchangeable, at the very least one would expect to be able to import building geometry from an upstream application that developed it or imported it successfully itself, and proceed using it without difficulty. This may not always be the case for one of the following reasons:

1. The data cannot be exchanged because, while they exist and are available, they are not part of the exchange "view" of the model.
2. The development of specific data is not part of the (design) problem for which upstream applications were used.
3. The data in question simply cannot yet be meaningfully defined at that stage of project development—it is too early in the project.

Exchange views limit data that are exchanged among participating applications to a subset of what is available. The end user is seldom aware of what is specifically excluded and expects all previously defined data to be available and importable. This is the case of the implementers' agreement to exchange data defined in the building model floor by floor. If the building has a multistory space (i.e., a space with height that exceeds one floor) complete information about that space is distributed over more than one floor, and is exchanged among interoperating applications accordingly. A downstream application that needs a complete definition of this space must "assemble" the information itself; if it cannot, some of the information is lost to it. Such an application must then rely on middleware that can pre-process the information and assemble the complete set before it reaches the application.

Some data may be perfectly exchangeable, but upstream applications did not create and insert them in the project model simply because the end users upstream did not concern themselves with these specific data. An example of that is the case where the building model does not include the equipment and furnishings in the immediate surroundings of the building, such as exterior lighting. End users who populated the project model with data about the building (in this case architects) did not create any data about exterior lighting because that is the task for which someone else is responsible (in this case lighting designers). A cost estimating application needs those data to generate a complete cost estimate. The end user who operates the cost estimating application must either invent the missing data and import them in the application externally, or find a way for the application to bypass its request for the data in question.

Inability to define meaningful data because it is "too early" in the project is a frequent issue. One of many available examples involves again cost estimating early in the building design process. Most cost estimating applications require detailed information about building foundations to generate a complete cost estimate. While the foundation system may already be known in schematic building design, it is too early for detailed foundation design – the necessary information that is needed in detailed foundation design has not been developed yet. The cost estimating end user must again either improvise the missing foundation data and import them in the application, or find a way for the application to bypass its request for those data.

LESSON 3: DATA INCOMPATIBILITY

Problems arise also from data incompatibility among interoperating applications. In some cases data defined by one application do not match exactly what another application expects. Or one application does the same type of operation differently from another, which results in data that require different interpretation and reformatting.

For example, CAD applications typically define individual walls as single instances, regardless of how many spaces such walls may be part of. Thermal analysis applications, on the other hand,

define walls as part of space; the single (long) wall defined in CAD may actually constitute a string of several (shorter) walls in the thermal analysis applications. The end user operating the thermal analysis applications must find a way to divide the imported (long) wall into multiple segments and "attach" them to corresponding spaces (thermal zones).

Some of the existing middleware can perform this task for the end user: associate wall geometry with defined spaces, generate geometry of wall segments, and attach them to appropriate spaces. This is not a trivial task; interior walls have spaces on both sides and often intersect with other walls. The problem gets even more complicated if floor and/or ceiling slabs are involved instead of walls, and the floor plan above does not match the one below. If the end user does not have access to such middleware, the building model has to be redefined to include the wall and slab segmentation needed by the thermal analysis application.

The problem in this particular example has its root in the original implementers' agreement forged by CAD implementers. For *valid* economic reasons related to the internal workings of CAD engines, the participating developers agreed not to implement space boundary. Thus, IFC compatible CAD applications do not presently populate project models with information related to space boundaries, leaving it to downstream software to resolve issues like the one in the example above. This will be rectified in the future with a new implementers' agreement reached in April 2002.

LESSON 4: BUILT-IN LIMITATIONS OF THE DATA MODEL AND APPLICATIONS

And then there are limitations that are built in the IFC object data model or in a given application. The IFC data model is far from complete. Several definitions critical to some of the industry professions are not part of the model yet, or have not been implemented in the specific software.

One critical inability is related to the modeling of dynamic exterior shading systems. Contemporary designs for glass buildings often employ sophisticated and intricate systems of exterior blinds, louvers and shading fins that protect the inside from excessive solar gain while permitting the penetration of natural light. It is not possible to define such systems in the current release of the IFC object data model, even if the CAD application can model them. The end user has no choice but to approximate them as something else, such as miniature beams and/or walls.

Some of the available interoperable software is not capable of generating some of the critical data that *can* be defined in IFC format. For example, some IFC compatible applications are incapable of defining or understanding geometry of pitched roofs. Other applications cannot deal with sloped or curved walls. In such cases the end user must redefine the geometry in question to mimic the actual geometry in a way the application can accept. In the case of downstream applications, this means that the end user must reach the CAD application upstream to reformulate the geometry.

LESSON 5: PROJECT MODEL EXCHANGE FILE SIZE

At this point all data exchange among applications involves the exchange of the entire project model, as defined by the particular exchange view. Even modestly sized buildings can result in very large project model files that can be tens of megabytes. While modern computer hardware can easily exchange files of that size, the manipulation of the data in the files can slow down the operation of some of the IFC compatible software to the point where it becomes annoying.

To be effective and avoid paging, any activity related to model checking and visualization requires a *large* RAM and video RAM; a *very* fast CPU and data buss can be of considerable convenience. Such hardware can be the difference in meeting project deadlines and avoiding frustration.

The IAI has started investigating partial model exchange. When a solution is eventually implemented, partial exchange promises to dramatically

reduce the size of future exchange files. End users of IFC compatible software will be have to rely on top-of-the- line hardware until then.

LESSON 6: CHOOSE SOFTWARE CAREFULLY

It does not help when the application is ridden with software bugs. Several IFC compatible software applications are still in beta status. While their developers usually respond promptly to bug reports and fix most bugs quickly, discovering and fixing software bugs slows down the work process and truly frustrates the affected end users. When end users continue to run into bugs or when developers do not fix them promptly, frustrated end users usually stop using the application, sometimes regardless of the investment they have already made in its use.

Industry software users have choices in selecting which software to use, even within the limited sample of interoperable software. When deciding which interoperable applications to deploy in a particular project, they should consider and thoroughly check *all* of the following:

1. Software functionality
2. Software maturity
3. Software's level of interoperability
4. Vendor support
5. Support from other software users
6. User's ability and skill to use all features of the software

The interoperable software application must be able to *perform all tasks* required in the project that it is expected to perform. That is the most important criterion in selecting a single application. Using software that cannot do all the tasks results in frustration, delays and excessive cost.

The software should be mature. It should work flawlessly, be free of bugs, well documented and be user friendly. Software that lacks these characteristics can also cause frustration, delays and excessive cost.

Some IFC compatible software supports more than one exchange view. Such software can typically interoperate with more applications of other type than the software that incorporates only a single exchange view. Using such software can simplify work on the project and can save time and resources.

Vendor support is critical. End user problems in data exchange that have not been foreseen and are thus not documented can arise even if the software is mature and is well documented. Vendor support is even more important for software that is less than mature. Poor vendor support causes frustration and delays, and can result in elimination of that application from the project.

The ability to obtain advice and tips from others that have experience in the use or in data exchange with the particular software can prevent many a headache and save time. The number of end users who can help others is still fairly small, but some help can already be found. As the use of interoperable software in industry projects grows, the body of experience and the pool of experienced end users will grow too.

End users should not overestimate their *own ability to use advanced features* of the candidate application(s). Many industry software applications require expert user knowledge and considerable experience in the use of the software to perform complex tasks they are capable of. "Learning on the job" is not the appropriate solution for end users who do not posses such expert knowledge or experience. Selecting a less sophisticated application or involving expert users in the running of the application is usually a more effective approach.

Of course, few currently IFC compatible software applications meet all the listed criteria. The end user must then strike a balance that works best for the project. Eventually, the selected software must be able to perform together as a *group* to accomplish project tasks more effectively and more efficiently than available applications that are not interoperable.

LESSON 7: INTEROPERABILITY IS INDEED BEGINNING TO WORK

This is possibly the most important conclusion from early deployment of interoperable software. Despite initial problems and limitations that are typical of "growing pains," software interoperability in the AEC/FM industry *is* beginning to work.

Deployment of IFC compatible software in pilot software deployment projects quoted above shows that gains can be made in industry projects even at this early stage of deployment. While only some of the deployment has been objectively observed (Kan, C., in prep.), actual gains have not been measured yet. This may not be an easy task to do; it will require the development of appropriate strategies and procedures of tracking and measurement.

Still, it is already evident that the use of three-dimensional building models that are prerequisite for software interoperability in the AEC/FM industry is changing the process of how buildings are designed, and how building components and systems are selected. No room is left for ambiguity in the creation of the building model; many design decisions that are typically made later in the standard design process now must be made up front when the model is generated. Design definitions of space above the plane of a given floor plan, as well as those in front or behind the plane of a taken building section, that are implied and left to the imagination of the viewer of two-dimensional drawings, now must be resolved entirely in 3-D. Models of buildings in early design stages require the consideration of much more detail than is normally included in 2-D drawings. This all leads to buildings that are much better thought out much earlier in the design process. In turn, this leads to fewer mistakes, conflicts, misunderstandings and corrections, fewer change orders later on.

Exchange of building geometry works flawlessly for the most part. Downstream applications in the three pilot deployment projects quoted above successfully imported available building geometry generated by CAD applications upstream, even if the acquisition of geometry was at first not as smooth as expected. Automatic or semi-automatic acquisition of building geometry dramatically reduces the input preparation for downstream applications, which should save significant time and considerable cost (Bazjanac 2001).

One of the emerging important benefits from three-dimensional building models is visualization "on the fly." Using a building model, different interoperable visualization software can relatively easily and quickly generate multiple high resolution images of different types, from different vantage points inside and/or outside of the building. Such images can play a significant role in effective communication about the building and encountered issues, as well as aid in model debugging.

LESSON 8: INTEROPERABLE SOFTWARE DEPLOYMENT REQUIRES TEAMS OF EXPERTS

Successful deployment of interoperable software requires *teams of experts*. These experts must have substantial experience and expert knowledge in the particular software as well as in the profession they represent. This may be the second most important message to this industry from early deployment of interoperable software.

An industry software end user who is a practicing professional may be an expert user of a particular software application, but seldom also possesses sufficient knowledge of the IFC object data model, the exchange view of the interoperable application, and the software's interface to the data model. At present, these are all required to smoothly operate interoperable software. A person with all these skills and knowledge is an expert.

The single end user, who wants software interoperability without the participation of others, must have that type of knowledge about all interoperating software used in the project. A typical industry end user is very likely to lack the necessary expertise in the use of any single interoperable

application (see LESSON 1 above), let alone multiple applications.

A different type of end user may be a software expert who is also in command of all the information listed above, but does not have the *professional expertise* that is needed to effectively operate the interoperable application. While possibly guiding a smooth data exchange, such a user may not be able to correctly apply the exchanged information.

At present, typical industry end users stand a very poor chance of reaping possible benefits from interoperability on their own. Individuals who possess both types of expertise extended to *multiple* interoperable applications are extremely rare. In absence of such individuals, it is necessary to assemble *teams* of experts who, *as a group*, can provide software and professional expertise for all interoperable applications that are deployed in the project. Only such teams of experts can facilitate a smooth exchange of data among multiple applications, avoid problems and find solutions quickly. Typical industry end users can find roles on this team, but should leave all interoperability issues to experts. This condition is not likely to change in the foreseeable future.

CONCLUSION

Software interoperability in the AEC/FM industry is beginning to work, though one can certainly encounter a number of annoying problems that need to be worked out. The benefits of interoperable software seem at hand.

While the IFC object data model in its present form is still far from complete, little seems to be wrong with the data model itself – its architecture, and the definitions of object/attribute/relationship and property sets that are included in the defined exchange views, serve their purpose well. In general, most technical problems encountered in the early deployment of interoperable software are related to the software itself or to the implementation of the IFC model.

Some problems will probably "fix themselves" with time. Software developers, given time, will certainly fix bugs in "mission critical" software and add needed functionality that is currently missing in some of the interoperable applications. The IAI will continue to extend the IFC object data model with new definitions, and those will be implemented by existing and additional applications.

Other problems are structural and will be considerably more difficult to solve. This includes filling in of the data that are missing in project models because upstream applications did not generate them, and cases where different interoperable applications are doing the same task differently.

The early lessons from interoperable software deployment should benefit both end users and software developers, as they are beginning to expose the strengths and the weaknesses of the available interoperable software, and the process of generating and exchanging data. This paper provides selected examples from actual software deployment cases in real-life projects to illustrate the lessons learned.

Some lessons are limited to issues of building geometry. This is by necessity, as by far most of the data exchange in the pilot deployment projects involved building geometry. Other lessons are general in nature or applicable to other types of data exchange.

At this point, most of the lessons result from and reflect deployment in early stages of building design. This too is by necessity. As buildings are constructed and start operation with the help of interoperable software, lessons will follow from early deployment of interoperable software in those stages of the building lifecycle.

WORK AHEAD

The deployment of interoperable software in this industry is only beginning. Much still needs to be done to facilitate this development.

Members of the AEC/FM industry will eventually *have to* become much more knowledgeable about information technology and software in general,

and specifically about software interoperability. The education of the industry must start now. It can start by educating a new generation of consultants and teachers who will teach future professionals: by finding a role for them in real life projects that are deploying interoperable software and involve some of the experts in interoperability.

The industry stands to reap many benefits from the use of interoperable software. The use of such software *should* dramatically increase and expedite the project information flow, reduce project and building delivery times, reduce overall project cost, and make building operation much more efficient and cost effective. It should result in better buildings. At this point such claims are still nothing more than claims. They need to be substantiated with measurement of actual benefits attained in real life projects.

Developers of interoperable software need to accelerate their effort. They often claim that investments in development cannot be justified without a beneficial and measurable effect of the investment on the market. But market acceptance of some of the newly interoperable software is not likely to markedly improve until that software becomes more robust. Software bugs must be fixed; planning for effective vendor support of a substantially larger end user population should start now.

In the future, project models that are populated by data from some of the downstream applications are likely to become extremely large. The IAI needs to solve the problem of partial data exchange. This will make the exchange of needed information that is contained in such enormously large data bases much more manageable.

ACKNOWLEDGMENT

This work was supported in part by the Assistant Secretary for Energy Efficiency and Renewable Energy, Office of Building Technology, Building Technologies Program, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098, and internal funding from Lawrence Berkeley National Laboratory.

REFERENCES

Backas, S. 2002. Comments during presentation, 23 April 2002, http://cic.vtt.fi/vera/IAI_Summit_2002/YIT.pdf.

Bazjanac, V. 2001. Acquisition of Building Geometry in the Simulation of Energy Performance. In R. Lamberts et al. (eds), *Building Simulation 2001, Proc. intern. conf., Rio de Janeiro,* Vol. 1: 305-311. ISBN 85-901939-2-6.

Bazjanac, V. 2002. Energy Efficiency and Electrical Reliability Laboratory: Virtual Building Design, http://cic.vtt.fi/niai/IAI_Summit_2002.htm.

Building Lifecycle Interoperable Software (BLIS) 2000. http://www.blis-project.org.

IAI Implementation Support Group (IAI_ISG). 2001. http://www.bauwesen.fh-muenchen.de/iai/iai_isg/.

International Alliance for Interoperability (IAI). 1995a. http://www.iai-international.org.

International Alliance for Interoperability (IAI). 1995b. http://iaiweb.lbl.gov.

International Alliance for Interoperability (IAI). 2002. http://www.iai-international.org/iai_international/Marketing/Pilots_List.jsp.

Iso-Aho, J. 2002. Benefits and Challenges in Using Product Model in Actual Capital Project, http://cic.vtt.fi/niai/IAI_Summit_2002.htm.

Iso-Aho, J. 2002. Comments during presentation, 23 April 2002. http://cic.vtt.fi/niai/IAI_Summit_2002.htm.

Karlshøj, J. 2001. Pers. comm.

Karlshøj, J. 2002. IFC Pilot Project - Headquarters for the Danish Broadcasting Corporation, http://cic.vtt.fi/niai/IAI_Summit_2002.htm.

Mok, J. 2002. Joint IT Effort between Government and Industry: A Unique Experience, http://cic.vtt.fi/niai/IAI_Summit_2002.htm.

Solibri, Inc. 2001. http://www.solibri.com.