



An Object-Oriented, Component-based Approach to Migrating Legacy Systems

Dr. Vladimir Bacovski

July 2004

An Object-Oriented, Component-based Approach to Migrating Legacy Systems

Dr. Vladimir Bacvanski
vladimir@inferdata.com

July 2004

Overview

Migration from a legacy to an object-oriented environment and component enterprise technologies is a difficult undertaking that faces numerous challenges. While some of the difficulties are of a technical nature, key challenges are in migrating development teams to new technologies, a new way of thinking and often to a new way of developing software altogether. This short white paper addresses the problems and the best practices in training teams who are migrating to the new technology, the associated thinking, and related process. The advice and the ideas in this paper represent state of the art methodologies and best practices used in successful adoption of the object-oriented and component development within the enterprise. The experiences are primarily from our work with a number of financial and insurance sector companies.

Changes in Thinking

Although the technical factors in migration often stand in the spotlight, the required change in thinking is a subtler, and yet critical, factor. Object-oriented and component technologies require different approaches to architecture and design than the traditional procedural software. Project failures, scalability and maintenance problems are common consequences of applying the new technology with the old-style thinking.

Key Factors

The change in thinking is needed in the areas of:

- **Design and implementation of software components.** New enterprise technologies are object-oriented and component based; traditional procedural paradigm is not suitable for modern component development. Yet, procedural is the entrenched way of thinking among legacy developers. Therefore, learning just the different syntax of the new programming language is not sufficient.
- **Software development lifecycle.** The change in technology is typically accompanied with a migration from waterfall to the iterative development. Changes in development and management practices must follow.
- **Applying the technologies.** The new technologies need to be carefully applied — merely using the new technology is not a guarantee for success.

Grasping the Big Picture

For the development team to be productive every member of the team needs to have a clear understanding of the overall development process, and how their peers use the results of their work. Otherwise, the developers will be creating deliverables that do not provide enough or the right kind of information to their peers. Example: business and system analysts who do not know how the use cases will be applied for architecture, design and implementation of the system will not be able to provide what architects, designers, implementers, and testers need to know from use cases.

The training needs to address the relationships between different artifacts and demonstrate the continuity of development from the domain to the implemented and tested system. This aspect of training is particularly important for financial and insurance companies that have a tendency to introduce a large number of narrowly defined roles.

The Myth of Documents and Process

An official adoption of a new development process that potentially demands excessive documentation is not a guarantee for success. We have witnessed organizations who prescribed creation of many documents, using new development processes, tools and technologies, yet paid little attention to improving the skills of their teams as they move from legacy to component development. A detailed process and a large set of documents can create an illusion of progress, but they are not a replacement for a team that has mastery over skills and technologies.

Through effective training and mentoring, the team needs to learn to competently apply the development techniques and use process and documents as a framework for thought process and productive development.

Overcoming Obstacles in Transition

The migrating organizations need to explicitly address the common obstacles that could make the transition less successful. The key challenges we have observed (predominantly at clients in the financial and insurance arena) are:

- **Team continues to think in the old way.** Problems with the legacy development approach will be migrated to new technologies if the team continues to think in the “old way.” InferData training is built from the ground up to change the way of thinking and applies the new technologies and processes in a realistic setting. The key in changing the thinking paradigm is to practice sound object-oriented and component design principles from the first day. The next step is coding and technology training. The alternative approach, where developers first learn the coding and then the good design process is only partially successful: the developers have a tendency to express the old legacy ways of thinking in the new language. When the design material is learned after the coding style was adopted, it can result in limited success.

- **Developers learn just the coding.** The training needs to address architectural and design issues and best practices of the new technology. Bits and pieces of low-level knowledge are not sufficient for building enterprise systems. In our preferred approach, developers' training begins with sound analysis and design techniques with objects through which developers change their way of thinking. Then, the developers learn the coding in the new technology — effectively expressing good designs in code.
- **Inability to effectively apply the new skills after the training.** InferData courses are designed to relate to real situations that teams will encounter in everyday development. Ideally, mentoring follows our training. The team has the opportunity to work with the mentor after the class, eliminating mistakes early and guiding the team hands-on while applying the new technology, tools, and processes. To achieve maximal effect, the training and mentoring should happen ideally immediately before the new project is started. Inviting a consultant to rescue a struggling project is proven ineffective.
- **Transition takes too long.** Slow transition training, paced over a long period of time is possible, but sub-optimal. If the team reverts back to old tasks between the training sessions, then the learning retention is put at risk and follow-up courses may waste time refreshing the old material. We have found a “boot camp” style to be a productive setting for training a range of skills needed in migration. The team will go through an intensive set of courses building practical skills upon each other, until the team can embark on the applications using the new technology. After the team members gain practical experience (ideally supported through mentoring), they can further improve their skills through advanced courses.
- **Single training curriculum does not fit all the roles.** Effective training needs to be targeted towards specific roles. InferData training paths deliver the knowledge and skills to specific roles in development, and can be further customized to address the needs of the organization. In addition, the courses that address legacy developers deal with the differences and similarities between the new and old technologies. As an example, our course “*Java for COBOL Developers*”, which targets legacy developers, is specifically designed for this audience and has numerous examples of mapping between the COBOL expressions and their modern equivalents and new ideas.
- **Teams are left on their own after training.** In our approach, the teams can continue working with their instructor as a mentor, helping to apply the learned material on the project. Also, we actively encourage and support forming and organization of special interest groups and learning communities within the client's company. InferData instructors will provide additional study material long after the class has ended.

The Role of Management in Transition

Clearly, the transition to new technologies and processes has an impact on management. Managing the projects in the waterfall style independently from the technological and process changes is not effective. Managers need to understand the practical consequences of

the new paradigm, new processes and technologies relative to their roles. This is accomplished through a particular learning path that addresses specifically managers. In order to get traction among busy managers training must offer a wealth of best practices and practical lessons learned that could be immediately applied.

Modernizing Legacy Projects

Most of the courses in the IT training marketplace are oriented towards building new systems from scratch. However, the reality is that companies need to integrate and modernize legacy applications with the new parts of the systems. Bridging the old and the new systems imposes particular challenges of organization in architecture, design and realization; these are left unaddressed in a typical curriculum. We believe that integration with legacy systems is not a subject that should be left untouched. Therefore, we train developers to apply effective architectural and design approaches for enhancement of the legacy system functionality, adding new features or web presence, and use of messaging or web service technologies as the integration vehicle. Also, the managers and development staff always appreciate a set of best practices and step-by-step recipes for modernization.

Aligning Business and Software Systems

Technology alone often results in a disappointing return on investment. In contrast, tangible benefits are achieved through software systems architected to directly support business. To realize this optimization, productive organizations should apply development process that starts with modeling the problem domain, capturing the system requirements directly aligned with the essence of the business, and finally is realized within a software system that is flexible and modifiable. Software systems built in this manner often enable substantive changes in business practices. In order to support this aligned view of software and business, our training approach applies the productive, modern thinking that firmly bases enterprise architecture in the problem and business domain. The alignment of business and software is both a strategic and tactical issue that we demonstrate throughout the whole curriculum.

Our Experience

InferData has been actively applying and researching the best practices of introducing object-oriented and component development to practitioners. We have been working with leading financial and insurance organizations in planning and delivering the migration. We are also actively advancing the training field: InferData was a co-organizer of the first workshop addressing the practical experiences of object-oriented education within industry and academia: “*Doing Your First Object-Oriented Project*”, held at the *Object-Oriented Programming Systems languages and Applications '97* conference. Our instructors were also invited to speak at the OOPSLA Educator’s Symposium several times. Our instructors and

mentors are actively involved in pursuing the state of the art in modernizing legacy systems: we were speakers at the *Object Management Group (OMG)* event in 2004 entitled *Architecture-Driven Modernization Workshop - A Model-driven Approach to Modernizing IT Systems*, with the talk “*Domain Driven Modernization of Legacy Systems.*”

Conclusion

Enabling legacy teams to successfully migrate towards object-oriented and component development needs to address a whole range of issues. This white paper points to some of the most critical topics that we address in our curriculum. Changing the way of thinking is the most pervasive issue; addressing details of new technologies is needed, but typically not sufficient. To achieve successful migration, legacy teams must progress through a carefully crafted curriculum in order to understand and effectively apply the new development paradigm, technologies and processes.