

**The RISE of Customizable Software
for the Factory Floor – Getting the perfect fit for your business**

Dr. Peter Green
BellHawk Systems Corporation

**Introduction**

Customizable software fills the gap between packaged software and custom software. This paper describes how customizable software can be used to rapidly implement factory-floor information systems using Rapid Implementation by System Evolution (RISE) methodology.

Classes of Software

Customizable software fills the gap between packaged software and custom software. The benefits and problems with each of these approaches is as follows:

Packaged software:

- One size fits all.
- Designed for widespread distribution in a standard form
- Usually well tested and debugged
- Configurable but not customizable.
- Have to adapt your operations and methods to fit standard package
- Often becomes bloatware, and hence hard to use, when features are added to satisfy a wide range of users.
- Interfaces limited to those built in to package.
- No software development needed to implement

Custom Software:

- Designed for specific application
- Very time consuming and expensive to develop
- Needs a lot of management time to decode what system should do
- Need to write comprehensive specification
- Needs very extensive testing to remove bugs
- Can be interfaced to all other systems
- High risk that system may not achieve desired results

Customizable Software

- Starts with a set of proven standard modules that are integrated into a working system that meets 70% to 90% of the requirements of the end users
- Modules are then customized to meet the exact needs of end users
- Can be interfaced to any other system.
- Can be customized to simplify user interface - avoiding bloatware issue.
- Quick implementation using RISE methodology.

RISE

The classic method of developing software is often called the "waterfall" method. It consists of the following steps:

1. Write and agree upon functional specification for system
2. Write and agree upon detailed specification for system
3. Code system
4. Test system
5. Integrate system
6. Deploy system

It is called the waterfall method because each step cascades into the next and they must be done sequentially. As a result, it can often take 18 months to 2 years to deploy even the first component of a factory floor information system. Because a factory-floor is a dynamic place that responds quickly to market requirements, systems implemented this way are often obsolete before they get deployed. In practice, the situation may be even worse, as the process often gets stuck on step 1. Managers, supervisors, and factory-floor workers engage in an endless round of meetings trying to decide what should be done until, in the end, everyone gives up in frustration and goes back to using pencil and paper systems.

An alternative to the waterfall method is Rapid Implementation Systems Evolution (RISE). With this method, you start with a solution that is proven to work but is not an exact match for your operations. You then have everyone, who is to use the system, evaluate the system and then write down what changes they would need or like. These are then divided into three categories:

1. Necessary
2. Beneficial but not essential
3. Nice-to-have

The necessary changes are then implemented using rapid cyclical development. In this:

1. The changes are made
2. The changes are tested by the users.
3. Users document further changes they need
4. Cycle back to step 1 until no more changes are requested.

If the budget allows, this is also done for beneficial and nice-to-have features. The system is then deployed, but the RISE process does not stop there. It is to be anticipated that changes will be required after deployment for the following reasons:

1. Improvements in can be made in operational procedures.
2. Can simplify user's job.
3. Need to make changes in manufacturing process
4. Need to add more modules to system.

Using RISE, given a good customizable software starting point, a factory-floor information system can be implemented in 30 to 90 days. Then changes can be made in a few days as needs arise.

Note how customizable software and RISE are linked. The customizable software provides the starting point for RISE. This may be 70% to 90% of the needed solution, but it has to be easily customizable to allow RISE to work. The waterfall method does not work with customizable software, as it is too easy to write-in a requirement that is incompatible with the customizable software, thereby precluding its use.

The success of RISE is based upon the observation that people can easily provide relative descriptions but have much greater difficulty with absolutes. If you ask people what color brown door they want for their home, they often have great difficulty describing what they want. Yet you show them a door, close to what they want, and they will tell you they want it lighter or darker, more or less grain, and so forth. The same applies to features people need in their software.

Note that RISE avoids writing lengthy specifications. The starting point software is the specification. Then all you have to do is specify the changes. In most cases, this should be done incrementally as people learn to use the system.

Customizable Software

Customizable software is, in many ways, a hybrid between packaged software and custom software. It is far more than a set of software modules. In its initially distributed form, customizable software works as an integrated system just like packaged software. The difference is that customizable software is designed to be easily customized.

So if I buy packaged software, and get the source code, do I have customizable software? The answer is maybe.

Buying customizable software is like buying a house in a development. You buy the model you want with the options you need. You then customize the house by painting and decorating and putting up shelves in the garage. You do not expect to have the builder replace your house with a later version of the same model when he comes out with one. No, you expect to live in that house until you trade it up for a newer or bigger house. In fact you would be very unhappy if, having just got your shelves installed and your tools hung up in version 3.1 of a garage, the builder came along and ripped it down to install version 3.2 of the garage.

Most packaged software vendors continuously come out with new versions of their software. In fact, the way they make most of their money is by charging you for new releases. If you have made changes to release 3.1, because you had the source, and the vendor comes out with release 3.2, you have to put your changes into the new release. As a result, manufacturers often find themselves stuck with unsupported old releases of systems because they were unwilling to keep up with a vendor's releases. In this case, you do not have customizable software, even though you have the source code, unless you are willing to forgo new releases of the system.

For front office systems, such as ERP, MRP, or accounting systems, sticking with packaged systems is generally a good idea as front-office operations are pretty standardized and new releases are often

beneficial. On the factory floor, however, customized systems are required due to the variety of operational activities, even by manufacturers making the same products.

Many ERP, MRP, and accounting package vendors have come out with factory-floor modules for their systems. Some have tried to overcome the customization barrier by providing the source code for their modules. They have then, however, invalidated the principles of customizable software by requiring the end users to update their factory floor module versions every time the front-office systems change.

In developing factory-floor information system software there are many tradeoffs. These tradeoffs are made differently depending on whether we are creating packaged software, customizable software, or custom software.

With packaged software, it is customary to distribute only the binary code. This software often has built-in license management mechanisms to restrict its use to only those customers who have paid for licenses. The source code for custom software is almost always given by its developers to the company that paid for its development. With customized software, some parts are distributed in binary form and some in source form. The source code modules are those that, in the experience of the developers, need modification from installation to installation. The binary components are those that are common across all potential applications of the software and may contain license management mechanisms.

The source code for packaged systems is whatever is easiest for the developers. For customized systems, it is whatever is easiest for the people who will customize the code. Thus in customizable software such as BellHawk you find a combination of Access and Visual Basic being used for the factory-floor PC clients. This is so that these can easily be modified by manufacturing engineers to fit the needs of the operational processes and to interface to process equipment such as weighing scales. You will also find extensive use of English-like IF...THEN... rules that can be modified by non-programmers for applications such as planning, scheduling, and rules-based alarming.

Packaged software is distributed in complete version updates. The goal with customized software is to be able to upgrade the factory-floor information system, not by replacing existing customized modules, but by "bolting on" new modules that add functionality. If the new features do not require changes to the existing software then this goal can be readily achieved. In some cases, where the existing features share screens with the new features, it is not possible to add the new feature without redoing customizations to the shared screens. These problems can be minimized by careful systems design. They can also be minimized through the use of rules to control actions.

Packaged software may come with a set of interfaces. In general, it is not possible to add additional interfaces. Customizable software typically comes with a set of generic interfaces that can be customized to specific applications. For example, customizable factory-floor software may support both batch flat-file and real-time ODBC/SQL data exchanges with ERP, MRP, or accounting systems. These generic interfaces can be customized for the specific system being interfaced to. Customizable software may support RS232 interfaces to weighing scales with protocols that can be customized for the specific weighing scales. It may also support flat-file exports to payroll systems that can be customized for data content.

Who Customizes the Software?

1. Customer's Manufacturing Engineers and Information Technology Programmers. They are able to extract the modules they have purchased licenses for from the installation disk and have them automatically installed on a server. They can then customize the client-side programs and install these on PCs and PDAs throughout the factory. They can also customize the interfaces to their existing systems.
2. Consultants and systems integrators who are certified by the supplier of the customizable software to be technically competent in customizing the software. These people perform the customizations when the manufacturer does not have qualified people available to perform this task. Consultants and systems integrators may also assist clients in deciding what customizations need to be made.

In both cases it is essential that the vendor of the customizable software provide:

- Well documented software.
- Good technical training courses.
- Good technical support.

Conclusion

Customizable software fills the gap between packaged software and custom software. It enables a customized system to be deployed rapidly using a RISE methodology. It is applicable when:

- Factory-floor information systems' requirements cannot be met by packaged software
- Custom systems development will take too long, cost too much or is too risky.

Customizable software can only be used, however, when the factory-floor system requirements can be met by the capabilities and architecture of the customizable software. Otherwise fully custom software development is the only viable approach.

For more information please call 1-800-747-1377, see our website at www.BellHawk.com, or send Email to sales@BellHawk.com.

Copyright © 2002 BellHawk Systems Corporation, an operating unit of Applied Real-Time Intelligent Systems Corporation. BellHawk ® and WebHawk ® are registered trade marks of BellHawk Systems Corporation