M ost requirements-engineering techniques and practices focus on eliciting requirements from existing, known customers. However, these techniques and practices aren’t sufficient for surviving in current highly competitive markets. In particular, small organizations typically lack sufficient resources to compete effectively with large companies. W. Chan Kim and Renée Mauborgne proposed the blue-ocean strategy to create new marketplace value and so make competition irrelevant. A “blue ocean” is a potential market where competition doesn’t yet exist. In contrast, a “red ocean” is a bloody market where companies compete intensely for a share of a limited market space.

How can you elicit software product requirements for creating new customer value and new markets instead of refining requirements of existing products? The blue-ocean strategy provides an opportunity for small organizations by creating new market spaces without competition. However, it doesn’t provide a specific process for developing a new product.

Value-Innovative Requirements Engineering is a novel requirements-engineering process that we developed to adapt the blue-ocean strategy for software organizations. The VIRE process helps investigate the potential desires and needs of existing customers as well noncustomers to create uncontested new market spaces. (See the “Related Work in Value-Added Engineering” sidebar for other approaches.)

Blue-ocean-strategy framework

The blue-ocean-strategy framework consists of a strategy canvas and four practical actions to systematically innovate value. The strategy canvas is an analysis tool that reveals the current state of participants in a well-known market. It also shows what customers gain from rival products in that market. The so-called ERRC practical actions reconstruct customer value perception by answering four questions from which the acronym derives:

- Which of the factors that the industry takes for granted should be eliminated?
- Which factors should be reduced well below the industry’s standard?
- Which factors should be raised well above the industry’s standard?
- Which factors should be created that the industry has never offered?

The blue-ocean-strategy framework also includes six principles. The first four formulate the strategy and last two execute it:
Related Work in Value-Added Engineering

W. Chan Kim and Renée Mauborgne have published several methods for formulating and executing a blue-ocean strategy.\(^1,2\) But a blue ocean is just a strategy: it provides the big picture but no specific processes, analysis procedures, and design methods. In particular, it lacks a method for transforming value-added customer requirements into real systems.

Several other techniques deal with this problem. Quality Function Deployment is a systemic method for tying design decisions directly to customer needs. Mural Erder and Pierre Pureur outline an approach to designing architectures that fully support QFD requirements.\(^3\)

Triz is an analysis method for formulating innovative ideas and systems.\(^4\) Joseph W.K. Chan and K.M. Yu applied it in a systemic approach for developing successful information systems.\(^5\)

Generally, cost-benefit analysis methods can handle cost and system quality trade-off problems.\(^6\) Ho-Won Jung provides an optimizing method that can lead to highest system quality against a given cost.\(^7\) Rick Kazman, Hoh In, and Hong-Mei Chen proposed methods to resolve conflicts among customer quality requirements,\(^8\) and Barry Boehm and Li Guo Huang have proposed a broad program of value-based software engineering.\(^9\)

Several of these tools and methods provide partial solutions but lack a holistic approach ranging from the reliable elicitation of value-innovative requirements to their realization in real systems. The ERRC decision matrix we adapted for the Value-Innovative Requirements Engineering process isn’t a ground-breaking tool. Rather, it provides a novel integration of best practices such as cost-benefit analysis and the win-win spiral approach for creating a blue-ocean market of value-added systems.

References


- Reconstruct market boundaries.
- Focus on the big picture, not on the numbers.
- Reach beyond existing demand.
- Get the strategic sequence right.
- Overcome key organizational hurdles.
- Build execution into strategy.

VIRE embodies this framework and its principles in software engineering requirements to assist in creating a new market, eliminating competition, and satisfying customers.

Value-innovative requirements

VIRE offers an integrated requirements-elicitation process to identify, analyze, and validate new customer values for creating a potential market. Its usefulness applies to

- developing a new product for creating a blue-ocean market,
- redefining existing requirements and eliciting a new market space, and
- combining more than two existing requirements to increase customer value.

Figure 1 shows the proposed VIRE operations concept. It begins with analyzing existing system requirements using the strategy canvas and building value-innovative requirements based on collecting stakeholders’ feedback, identifying customer values and their conflicts, and analyzing existing systems and market situations. The strategy canvas is a tool for examining the current value proposition and creating new customer values. The ERRC analysis provides guidelines on which components to eliminate, reduce, raise, and create in achieving the ultimate goal of value maximization. VIRE also includes a quantitative, axiomatic validation method to check the degree of coupling among the redefined components.

VIRE process overview

VIRE consists of five steps adapted from the win-win spiral model.\(^1\) All steps must iterate at least three times. The first iteration redefines requirements, the second iteration focuses on building a prototype, and the third on product development.

Each step has inputs, outputs, constraints, and enablers. Some outputs from the previous step become inputs to the next step. In addition, each step can feed back to the previous step. The pro-
I E E E  S o f t w a r E

www.computer.org/software

cess finally outputs novel requirements for developing value-added products.

Step 1: Defining goals. The first step analyzes project goals, system boundaries, and system-value propositions. The analysis is based on a customer-perception survey of existing systems. It uses the strategy canvas as a diagnostic tool and action framework for building a compelling blue-ocean strategy. The strategy canvas serves two purposes: it captures the known market space’s current state, and it identifies what customers receive from the existing competitive offerings.

VIRE captures the strategy canvas in a graphical form, in which the horizontal axis captures the factors the industry competes on and invests in and the vertical axis captures the offering level that buyers receive across all these key competing factors. We present an example later with our case study.

Step 2: Identifying new values. To identify new value factors ignored in the past, step 2 elicits customer needs through an extended customer survey. The survey includes noncustomers who might use the product if its value proposition changed. Newly identified value factors are added to the strategy canvas developed in step 1.

Step 3: Analyzing ERRC. For resolving value conflicts among customers and prioritizing values within the available development resources, the ERRC analysis helps stakeholders eliminate, reduce, raise, and create for value innovation.

Step 4: Redefining requirements. This step redefines requirements for a value-innovative product on the basis of the newly identified and resolved values.
from steps 1, 2, and 3. The result is a new strategy canvas.

**Step 5: Validating requirements.** By comparing the old and new strategy canvases, stakeholders can intuitively understand the value-proposition changes and validate the redefined requirements. However, VIRE employs an axiomatic approach to validate the requirements in a more formal way.

**ERRC analysis**
Steps 1 and 2 identify new customer values, but the values aren’t yet harmonized to define value-innovative products. In step 3, the ERRC decision matrix that we developed for VIRE helps fine-tune customer values. Figure 2 shows the matrix’s five components:

- **Customer requirements (CR)** satisfy customer needs as defined in step 1.
- **System elements (SE)** realize customer requirements.
- **Customer importance (CI)** prioritizes the customer requirements identified in step 2.
- **Correlations (R)** reflect intersections between customer requirements and system elements.
- **ERRC decisions (ED)** reflects business importance (BI) and relative cost (RC) of system elements as well as suggested ERRC actions to eliminate, reduce, raise, or create their values.

We apply the ERRC decision matrix in three steps.

**Step 3.1: Prioritize customer values.** We investigate CRs, SEs, and CIs and enter the result in the ERRC decision matrix. As figure 2 shows, we rate CI on a scale from 1 (not important) to 5 (extremely important). Sorting CI yields an initial prioritization of customer values, but VIRE also considers resources such as development costs and available system elements, which temper the initial CI sort.

**Step 3.2: Map requirements to system elements.** We validate the prioritization in step 3.1 by mapping CRs to SEs and thereby derive their correlations, R. We use R to determine the business importance (BI) for each SE—that is, whether it eliminates, reduces, raises, or creates value. We assess R using the 9-3-1 relation-matrix strength rating, where 9 reflects a strong correlation, 3 reflects a moderate correlation, and 1 is a weak correlation.6

**Step 3.3: Determine customer values.** The ERRC matrix suggests ERRC actions by calculating BI and relative costs (RC), according to the following equations:

\[
BI_j = (CI_1 \times R_{1,j}) + (CI_2 \times R_{2,j}) + ... + (CI_n \times R_{n,j})
\]

\[
RC_j = \left( \frac{\text{Cost of SE}_j}{\text{Cost of } \sum_{j=1}^{n} \text{SE}_j} \right) \times 100
\]

No perfect algorithm exists for selecting the ERRC actions to apply for all projects. We initially considered both a return-on-investment approach and a win-win approach, but finally opted for a hybrid.

The ROI approach, first sorts SEs by the ROI value, where \(ROI = BI_j / RC_j\). Then the stakeholders decide on a threshold value for including or excluding SEs in a set of new requirements—for example, if \(ROI_j\) is less than or equal to the threshold value, SE\(_j\) is excluded. If a newly suggested SE\(_k\) is greater than the threshold, SE\(_k\) is added to the requirements; otherwise, it’s eliminated. Stakeholder discussions can increase or decrease the ROI for some important SEs.

In the win-win approach, stakeholders negotiate which SEs to include or exclude on the basis of each SE’s value proposition. The win-win spiral approach includes a framework to support these group negotiations.5,7

Figure 3 illustrates the VIRE hybrid approach. First, we identify the state of SEs according to the ROI analysis. Then, from the analysis results,
we use the win-win approach to consider stakeholders’ values and decide on the ERRC actions. In figure 3, the ERRC actions determine the areas marked “Eliminate,” “Reduce or raise,” and “Create.”

Axiomatic requirements validation

The VIRE process uses an axiomatic method to validate product design. The method maps what customers require to how to achieve it in a system domain (see figure 2). A set of CRs in the customer domain is mapped onto the system domain using a suitable set of SEs. The mapping process must satisfy design axioms, including the independence axiom: maintain independence in functional requirements.

After defining the functional requirements and SEs, the system developers use an equation such as the following to define a map between the customer and system domains:

\[
\{\text{CRs}\} = [R] \cdot \{\text{SEs}\}
\]

where [R] is a correlation matrix.

To satisfy the independence axiom, [R] must be either a diagonal or a triangular matrix. A diagonal-matrix design is called uncoupled design, as shown in equation 1.

\[
\begin{align*}
\{\text{CR}_1\} &= \begin{bmatrix} R_{11} & 0 & 0 \\ 0 & R_{22} & 0 \\ 0 & 0 & R_{33} \end{bmatrix} \cdot \{\text{SE}_1\} \\
\{\text{CR}_2\} &= \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \cdot \{\text{SE}_1\} \\
\{\text{CR}_3\} &= \begin{bmatrix} R_{11} & 0 & 0 \\ 0 & R_{22} & 0 \\ 0 & 0 & R_{33} \end{bmatrix} \cdot \{\text{SE}_1\} \\
\end{align*}
\]

In an uncoupled design, a single SE satisfies each CR. The result is an ideal system design with no potential conflicts between CRs relative to the SE.

A triangular matrix design is called a decoupled design, as shown in equation 2.

\[
\begin{align*}
\{\text{CR}_1\} &= \begin{bmatrix} R_{11} & 0 & 0 \\ R_{21} & R_{22} & 0 \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \cdot \{\text{SE}_1\} \\
\{\text{CR}_2\} &= \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \cdot \{\text{SE}_1\} \\
\{\text{CR}_3\} &= \begin{bmatrix} R_{11} & 0 & 0 \\ 0 & R_{22} & 0 \\ 0 & 0 & R_{33} \end{bmatrix} \cdot \{\text{SE}_1\} \\
\end{align*}
\]

In a decoupled design, some CRs affect other CRs, but we can resolve the conflict by adjusting an SE that has no relationship with another CR.

A design matrix that’s neither diagonal nor triangular is a coupled design, as shown in equation 3.

\[
\begin{align*}
\{\text{CR}_1\} &= \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \cdot \{\text{SE}_1\} \\
\end{align*}
\]

In this case, no independent SEs exist for resolving conflicts between CRs.

Coupled designs are unsatisfactory in the axiomatic approach because they can’t meet the independence axiom. The coupling originates in an inadequate mapping of CRs to SEs or in conflicts among the CRs. In this case, we suggest redefining CRs and SEs to establish a diagonal or triangular matrix that can resolve the conflicts.

A case study

We’ve applied the VIRE process to several real-world projects, such as a camera phone, a Web search system, and an information system for the Korean national tax service. Here, we show how the tax-service system created significant new value beyond the project’s original demands.

The project aimed to develop a public-auction system for the national tax service by adapting a public-sale-management system that the government had used since 1997. A small team (about six members) managed the VIRE process.

Step 1: Defining goals

The project aimed to provide new value for existing tax-service customers and attract new customers as well. To define project goals, the team first examined existing systems with respect to customer value. Team members surveyed noncustomers as well as existing customers. The red line in figure 4 shows the strategy canvas developed to identify the existing system’s status and problems. The results indicate participant dissatisfaction with the existing system’s performance, operability, work suitability, and functionality. Participants expressed mainly satisfaction, however, with its processing speed, security, and effectiveness using memory.
Step 2: Identifying new values

The team elicited all customer requirements and selected the 15 highest-priority requirements, shown in table 1. The ERRC analysis raised and created requirement values that would ensure customer satisfaction. The requirements included adding new tools and processes (CR15), an electronic approval system (CR2), and glue code (CR3). These functions are potential factors for increasing customer value. Team members analyzed the priorities and customer importance from the requirements. Analysis results showed that changing the existing (closed) systems to secure open systems was the most important factor to customers. An automatic-alarm function proved unimportant to them, so it ranked lowest among the requirements.

Step 3: Analyzing ERRC

The project team related SEs to CRs and analyzed SE costs from the requirements. Figure 5 shows the SE hardware or software required to realize customer requirements and the relative costs to implement SEs to satisfy their correlated requirements. In the analysis of 15 SEs, system software (SE1) represented 28.9 percent of the entire system cost. Next, business software and the relational database (SE10) represented 26 percent. In contrast, SE12, SE14, and SE15 represented just 1 percent of the entire cost of the system.

The team calculated the importance customers placed on the 15 requirements by scoring the prioritized requirements identified in step 2. The CI value is therefore based on the CR priorities. High-priority requirements received high values. In our case study, CR1 and CR4 were degree 5, which is the highest value; CR3, CR4, and CR11 were degree 1, which is the lowest.

The team calculated business importance using CI values and correlations according to the equation in step 3.3: the BI of SE1 is multiplied by CIj with Rij for all CRs involving SE1, then all Rij values are summarized.

Applying the ERRC hybrid approach, the project team decided to create four new system elements based on customer requests—specifically, SE3, SE10, SE13, and SE15. The team also eliminated three major system elements—SE2, SE3, and SE5—in response to stakeholder negotiations. Moreover, the ERRC analysis results supported reducing the performance of SE3 and SE8 and increasing the performance of SE9 and SE12.

Step 4: Redefining requirements

The green line in figure 4 shows the new strategy canvas, which reflects the ERRC analysis results and revised requirements. The project team used the data analysis in step 3 to verify the strategy canvas according to the project goal.

Step 5: Validating requirements

The team used the axiomatic approach to validate requirements. The initial matrix wasn’t verified, and some conflicts showed up among the CRs and SEs. The team solved the problems by transforming the initial matrix through recoupling, regrouping, and relocating CRs and SEs. When a conflict appeared, the team performed an ERRC analysis and changed the system architecture. The system architecture resulting from the VIRE process therefore reflects iteration and redefinition of requirements until they verifiably reflect the proper SE selections.

### Table 1

<table>
<thead>
<tr>
<th>CR no.</th>
<th>Customer requirements</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR1</td>
<td>The new system must be linked indirectly with the tax-service system (host) because of security problems.</td>
<td>1</td>
</tr>
<tr>
<td>CR2</td>
<td>A supporting tool for communicating between the national tax-service system (host) and the new system (Unix) is required.</td>
<td>2</td>
</tr>
<tr>
<td>CR3</td>
<td>The system must allow headquarters to examine the statistical data.</td>
<td>13</td>
</tr>
<tr>
<td>CR4</td>
<td>The system must implement additional functions, such as preserving attachments prior to national tax establishment.</td>
<td>14</td>
</tr>
<tr>
<td>CR5</td>
<td>A glue code is required to increase the fitness of the relative data.</td>
<td>8</td>
</tr>
<tr>
<td>CR6</td>
<td>The database of the national tax-service system (host) must be linked to the database of the new system (Unix).</td>
<td>6</td>
</tr>
<tr>
<td>CR7</td>
<td>The system must be linked to the national tax-service system and to an electronic approval system.</td>
<td>4</td>
</tr>
<tr>
<td>CR8</td>
<td>The systems must have real-time monitoring of the progress of public auction.</td>
<td>9</td>
</tr>
<tr>
<td>CR9</td>
<td>The system must tolerate the load of 2,000 users at the same time.</td>
<td>3</td>
</tr>
<tr>
<td>CR10</td>
<td>The database (file system) must be changed to a relational database to improve speed and data management.</td>
<td>10</td>
</tr>
<tr>
<td>CR11</td>
<td>An alarm function is required to avoid missing important jobs.</td>
<td>15</td>
</tr>
<tr>
<td>CR12</td>
<td>The system must keep a stable speed while the court auction information is entered.</td>
<td>12</td>
</tr>
<tr>
<td>CR13</td>
<td>The system must be implemented with consideration given to maintainability and extendability.</td>
<td>7</td>
</tr>
<tr>
<td>CR14</td>
<td>The screen must be designed to make it highly usable.</td>
<td>11</td>
</tr>
<tr>
<td>CR15</td>
<td>The business-process-management solution must be aware of what users do in each process.</td>
<td>5</td>
</tr>
</tbody>
</table>
**Evaluation results**

To determine whether VIRE-based developed systems created a new market, we adapted an observational method suggested by Alan Hevner and his colleagues. This method served as the theoretical base for comparing the two national tax-service systems—namely, the old, traditionally developed Asset Management System and the new, VIRE-based Public Sale System. As table 2 shows, the new PSS logs about 19,000 users for 6 hours a day compared with the AMS’s 2,000 users for 5 hours a day.

The development costs weren’t dramatically greater, even though the project added new elements to create new customer value. This is because the project eliminated low-value requirements. In addition, the development time was shorter, even though the PSS requirements analysis period was one month longer than AMS’s.

Eliciting requirements from not only existing customers but also noncustomers, such as senior managers and other agents, was a critical success factor. Noncustomers helped identify user interface and system functional problems to correct. These improvements helped PSS win a “best project” award from the Korean government in 2006. Giving stakeholders and noncustomers incentives to become involved early in the project early—for example, by explaining benefits of the new systems to them—was another important lesson learned.

In short, VIRE added new customer values in the PSS and created a new market space. PSS was successful enough to open a new market space—a blue ocean—in the Korean government information system domain.

---

**Figure 5. ERRC analysis results for Korean national tax service.**
### Table 2
Comparing VIRE with the Traditional Method

<table>
<thead>
<tr>
<th>Category description</th>
<th>Traditional method</th>
<th>VIRE method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project name</td>
<td>Asset Management System</td>
<td>Public Sale System</td>
</tr>
<tr>
<td>Cost</td>
<td>US$0.8 million</td>
<td>US$1 million</td>
</tr>
<tr>
<td>Increased users</td>
<td>2,000–2,000</td>
<td>2,000–19,000</td>
</tr>
<tr>
<td>Mean connecting time</td>
<td>5 hrs.</td>
<td>6 hrs.</td>
</tr>
<tr>
<td>Leading users</td>
<td>Clerks in only the national tax and asset management office</td>
<td>Clerks and middle administrators including other agencies</td>
</tr>
<tr>
<td>User satisfaction</td>
<td>60% positive</td>
<td>80% positive</td>
</tr>
<tr>
<td>System success</td>
<td>Maybe</td>
<td>Yes</td>
</tr>
<tr>
<td>New market exploration</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### References


### Acknowledgments

The Ministry of Information and Communication, Korea, supported this research under the Information Technology Research Center support program, supervised by the Institute of Information Technology Advancement (IITA-2006-C1090-0603-0032). For further information, contact Hoh P. In at hoh_in@korea.ac.kr.

### About the Authors

**Sangsoo Kim** is a PhD candidate in the College of Information and Communications at Korea University and an aircraft maintenance officer (Major) in the Republic of Korea Air Force. His research interests are software engineering economics, requirements engineering, self-adaptive software, and modeling and simulation. He received his MS in modeling and simulation from Korea National Defense University. Contact him at the Embedded Software Engineering Lab., Dept. of Computer & Communications Eng., Korea Univ., 5-ga Anam-Dong, Seongbuk-Gu, Seoul, 136-713 Korea; sookim@korea.ac.kr.

**Hoh Peter In** is an associate professor in the College of Information and Communications at Korea University. His primary research interests are requirements engineering, value-based software engineering, situation-aware middleware, and software security management. He received his PhD in computer science from the University of Southern California. Contact him at the Dept. of Computer & Communications Eng., Korea Univ., 5-ga Anam-Dong, Seongbuk-Gu, Seoul, 136-713 Korea; hoh_in@korea.ac.kr.

**Jongmoon Baik** is an assistant professor in the School of Engineering at Information and Communications University. His research interests include software process modeling, the software testing process maturity model, value-based software engineering, software economics, software reliability engineering, and software six sigma. He received his PhD in computer science from the University of Southern California. He’s a member of the ACM and IEEE. Contact him at the School of Eng., Information and Communications Univ., 119 Munjiro, Yusong-gu, Daejon, 305-732, Korea; jbaik@ic.ac.kr.

**Rick Kazman** is a professor in the Department of Information Technology Management at the University of Hawaii and a visiting scientist at Carnegie Mellon University’s Software Engineering Institute. His research interests include software architecture, design and analysis tools, software visualization, software engineering economics, human-computer interaction, and information retrieval. He received his PhD in computational linguistics from Carnegie Mellon University. Contact him at the Dept. of Information Technology Management, Shidler College of Business, Univ. of Hawaii at Manoa, 2004 Maile Way, Honolulu, HI 96822; kazman@hawaii.edu.

**Kwangsin Han** is a project manager in information systems development at Kamco (Korean Asset Management Corp.) in Seoul. His research interests are requirements engineering and software project management. He received his MS in software engineering from Korea University. Contact them at the Dept. of Information Systems, Kamco, Asia-Europe Meeting Tower 28F, World Trade Center, 159-1, Samsung-Dong, Gangnam-Gu, Seoul, 135-798 Korea; kshan7@kamco.or.kr.