Expanding Capabilities of Legacy Systems through Requirements Validation

Transitioning from Requirements Driven to ROI Driven Solutions

David Cohen, Gary Larson, Doug McDougal and Bill Ware
david@sente.com, gary@sente.com, doug@sente.com, bill@sente.com
sente.com Corporation
7209 Briarnoll Dr., Dallas, Texas 75252-6356 USA

Abstract

IT organizations are increasingly expected to provide business solutions faster and with better return on investments (ROIs). The strict technology focus on software features fails to deliver predictable, relevant and cost effective business solutions. Review of several order management applications developed over the last 20 years shows that software solutions are not able to respond effectively to evolving business-needs, becoming irrelevant to the business.

*The electronic customer contact management (**eccm**) toolkit has been used to validate legacy systems' requirements reducing the churning of functionality caused by incomplete and incorrect requirements, while extending system life cycle and reducing maintenance costs. The **eccm** toolkit tackles this business challenge by delivering:*

*1. **Validated software requirements** in areas such as business rules, web based user interfaces, operational processes, architecture, configuration management, and center management.*

*2. **A predictable methodology that delivers effective business solutions vs. traditional specification based software.** This approach enables software organizations to estimate and/or guarantee the business-solution's ROI in spite of the predictably incomplete and incorrect requirements that will be provided by the business unit.*

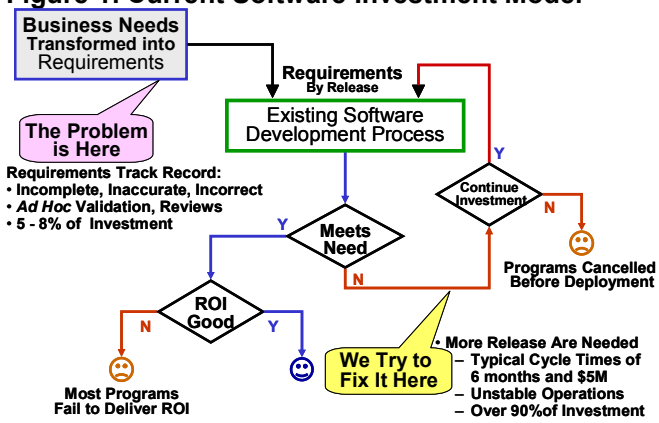
Keywords: *Software Engineering, ROI, Systems Engineering, Legacy Systems, and Center Productivity.*

1. Investment in Software Solutions

Businesses find themselves today under significant economic pressures to deliver products/services that generate new revenue streams. As a consequence, IT organizations are expected to provide business-solutions faster and with better ROIs. The limited **software focus** of these organizations fails to deliver predictable, relevant and cost effective *business solutions*. This has reached a crisis level in the communications industry. The process responsible for translating the business needs into requirements specifications is ad hoc and without well-defined success criteria for either completeness or accuracy [1]. This is a major challenge for two primary reasons:

1. There is great difficulty in translating high level business needs to detailed requirements specifications that drive the development process (e.g., increase share price by $2.50, improve the yearly productivity of a Rep by 35% in throughput and 20% in quality, improve cycle time of product delivery by 20%). The business needs will evolve over time and the systems are expected to adjust. This becomes a challenge in a highly competitive market place that demands faster and faster solutions that cannot be met with incremental releases (e.g., annual or semi-annual) unless the software has the built-in capability to adjust *without* software development. These are referred to as **e-capabilities** or **E-Type** based systems, with expanded configuration management (CM) features. This is further complicated by the limited communication between the business and the technical staffs; *usually* the technical team does not have an understanding of the effective business use of the newly planned capabilities. Finally, the requirements are always specified under significant schedule pressures because they are a key bottleneck for budget and development planning [2, 3, and 4]. **Figure 1** shows that business solutions delivered through this investment model are being "debugged" through *slow* and *expensive* development cycles making ROI delivery impractical (e.g., there is a typical 1:20 investment ratio between the task of requirements specifications vs. development).

Figure 1: Current Software Investment Model



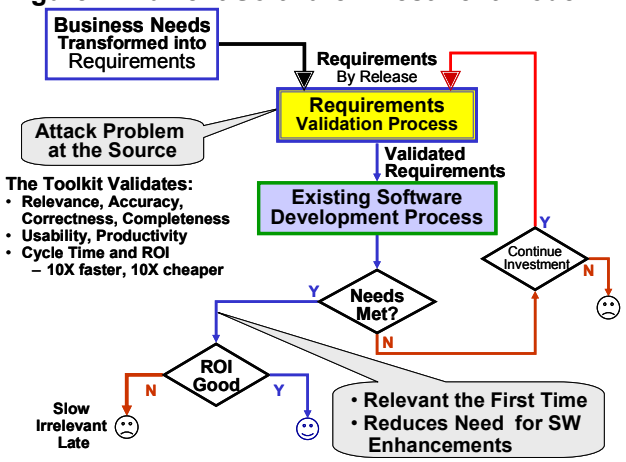
2. The conventional **specification-driven** (**S-Type**) investment paradigm incorporates the **wrong economic incentives** to deliver a business solution cost effectively. The business unit is responsible for both *specifying the requirements* and *ROI delivery*. The IT organization is responsible only for delivering quality *software features* based on *these* requirements. The limitations of the resulting solution are clearly the business unit's "fault", creating an economic model that rewards the IT organization (e.g., vendor) with *additional development resources* while it continues to deliver software irrelevant to the business **with limited/no ROI**. This is the primary reason that even system integrators (e.g., Accenture, Cap Gemini, IBM, EDS), with extensive business knowledge, prefer to deliver ***software*** and ***not solutions***. **It is therefore not surprising that in all projects the requirements continue to be incomplete, inaccurate and wrong.**

Analysis of over 250 projects shows that in initial implementations of medium size projects only 5% of the requirements were correct; in large projects only 1% of the requirements were correct [7]. This systems engineering phenomenon is referred to as **software pollution**™; it produces initial specifications that are both incomplete and inaccurate. **Software pollution is the principal reason why software investments consistently deliver limited or no ROI.** The traditional corporate view has been that IT is a standalone cost center. Although its basic mission is understood,

executives typically have not demanded the same business/ financial accountability from IT as they have from other departments. This is due to several factors including lack of executive technical background, intimidation by "techno talk" and, primarily, misunderstanding the difference between an IT work effort and an IT business product. However, today's economic climate does not allow such laissez faire oversight from either executive level or the funding organization. Timely, relevant, ROI driven solutions *with measurable benefits* (e.g., improve monthly productivity of a Rep by 35% in throughput and 20% in quality) must be demanded from IT organizations. **Transitioning the software business from a requirements-driven to a ROI-driven operation** [10, 11] will not happen without a joint commitment of executives and funding business units.

Figure 2 presents an investment model developed in support of this transition. The development process is driven by *validated requirements* producing relevant solutions the first time, with predictable and measurable benefits. The total software development effort is reduced significantly, because we no longer have to use *slow, expensive and painful* development-cycles to identify the "true requirements" of the business solution [6, 7].

Figure 2: Current Software Investment Model



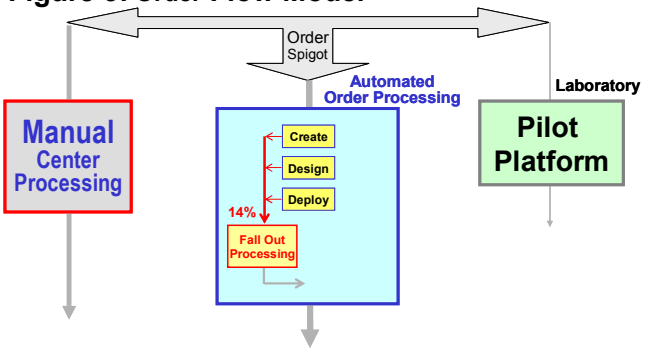
The ***eccm*** toolkit supports the requirements validation process in six key areas of an order management solution: architecture, configuration management (CM), web based user interface (UI), business rules (BRs), operational processes and center productivity management. The toolkit was incrementally expanded to include over 400 *e-capabilities*, allowing **faster** design changes during the validation process without software development (e.g., CM capabilities). This produced a secondary benefit, an earlier detection of issues that were somehow missed and not included in the project plan. Surprises that show up late in every program are key contributors to deployment delays and cost overruns. This is the primary reason why our development team can **profitably deliver timely business solutions with predictable ROI** using the incomplete input provided by the business unit.

2. Order Management Applications

A recent review of a ~$3B investment in order management solutions over a period of 20 years was found to include more than six different implementations. *The business challenge is to provide an environment in which we can effectively exchange complete and correct orders with customers.* Even today, many orders are delivered with limited quality control over transaction content (e.g., mail, fax, email) resulting in delayed and unpredictable order processing.

Figure 3 shows an economic order-flow model for the incremental migration of order-processing from a manual to an automated solution. This is referred to as order **flow through** (FLT) with limited/no human interaction. The assumption is that a software-only solution is more cost effective because of the fragile and unpredictable nature of manual processes. The model must also include a laboratory environment (e.g., Pilot) where each new software release is validated for its FLT level per transaction type (e.g., 80% or higher).

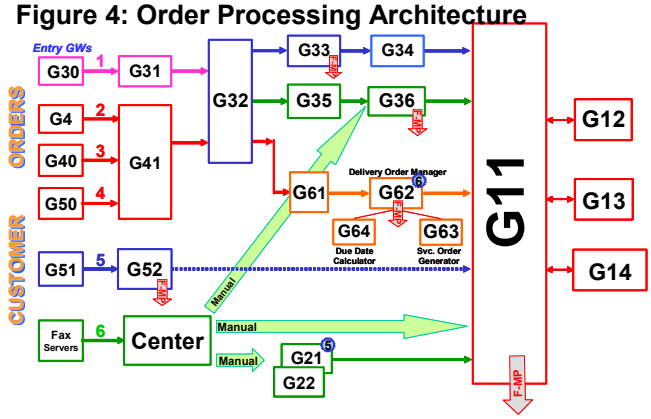
Figure 3: Order Flow Model



Orders that do not FLT automatically are referred to as *fallout.* Fallout order processing is less efficient than manual processing because it is difficult to process effectively *unpredictable* or unplanned activities. For example, *it is difficult* to make 100 trained Reps available in response to unexpected 3000-order fallout. Therefore, deployment of low FLT solutions results in limited/no ROI.

Figure 4 shows the evolution of the architecture that consists of over 6 generations of systems (e.g., G1 through G6) using a variety of technologies and development organizations. Initially, order processing required the order data entry into a set of *legacy systems* such as service order systems (G11) and billing systems (G12, G13, G14). Each system validates the order information for

Figure 4: Order Processing Architecture



both accuracy and completeness (e.g., telephone numbers, addresses, contract information). In some cases error processing may require contacting the customer for clarifications ("clarifies") to ensure the accuracy of the data. These "clarifies" are transmitted to the customer using telephone calls, faxes and/or emails, interactions that are difficult to track effectively. Similarly, customers unhappy with the progress in their order processing are provided the option, through "escalations," to increase the processing priority for a particular order. Clarifies and escalations reduce significantly the productivity of center Reps.

The initial main-frame (M/F) architecture (G**1**X), was expanded to support additional M/F functionality to handle new order types and EDI (G**2**X, G**3**X), with a mini-computer based EDI solution (G**4**X) to web based solutions (G**5**X, G**6**X).

Because of the large investment in computer-to-computer interfaces, customers are allowed to continue using older solutions producing an expanding system maintenance effort. A closer examination shows that most system-changes occurred either on the *customer access* side (G30, G40, G50) or the *order generation* side (G21, G22, G34, G36, G62) while the functionality of the original G11, G12, G13, and G14 systems continues to be an integral part of the solution.

All attempts to reduce maintenance cost by replacing these G1X systems through the use of more recent technologies have failed.

The architecture clearly shows that orders are still being accepted through a set of fax servers and email, and processed manually by the center Reps using G11 directly or through order generators G21 and G22.

Center productivity has not kept up with the demand for order processing causing center staff increases in line with order volume growth (e.g., 30% per year). This is caused mainly by the *limited order tracking capability* in the center combined with an ever-increasing *number of escalations* that results in a crisis mode of operation (e.g., high stress levels and reduced Rep productivity). On the average it takes six months to train a newly hired Rep, creating a major recruiting and training challenge.

Most software investments have failed to improve Reps' productivity and failed to deliver promised ROI. Specifically, investments in automation delivered transaction FLT levels of 35%-40%. As long as the solutions *accept incomplete orders* and software releases are not validated for their *robustness before deployment*, **ROI improvements are not feasible**. During the last five years, the regulatory environment combined with increased competition necessitated the rapid introduction of new capabilities and services. During this period of time we accumulated a **backlog of 1767 development features**. Each backlog feature is a by-product of *the delivery of new, incomplete and incorrect features*.

Figure 5 shows the feature development backlog as a function of the year the item was opened. There are still 985 items open since 2001 and 379 from 2000. *There are still 9 items open since 1997.*

Figure 5: Feature Development Backlog

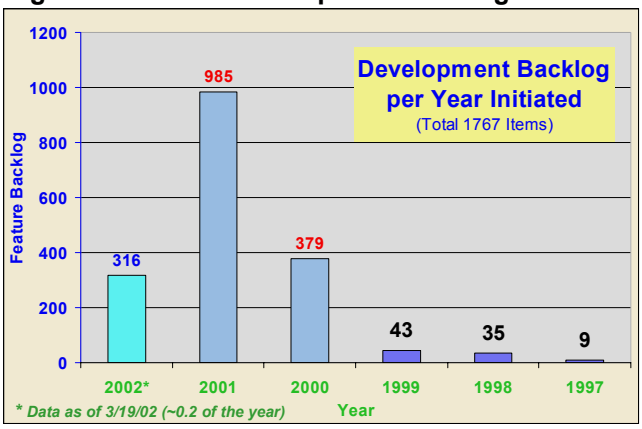
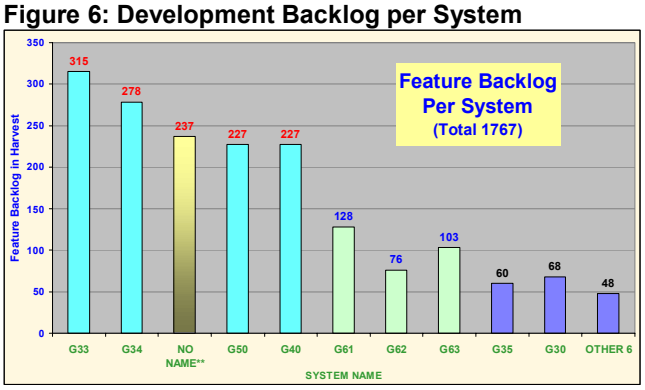


Figure 6 shows the distribution of the same information by system. The data shows that G**3**3, G**3**4, G**4**0 and G**5**0 have the largest backlog independent of *implementation technology* and/or *development team*.

The "No Name" system represents database tracking errors, for 237 items we could not identify the system association. The newest systems (G**6**X) demonstrate no improvement in their backlog rate over older systems; the smaller backlog is caused primarily by a recent two-year application deployment.

Figure 6: Development Backlog per System



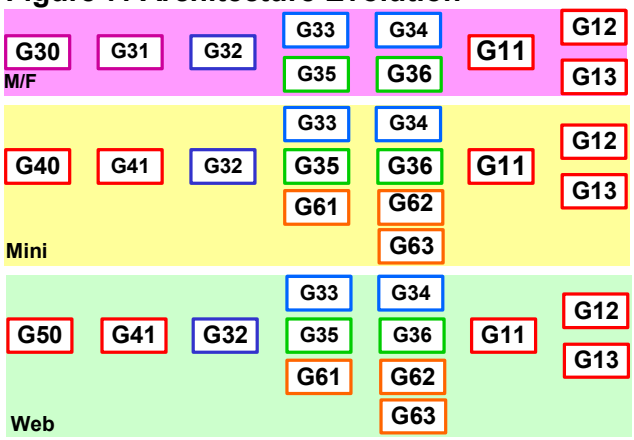
3. S-Type and E-Type Software Solutions

The need to meet the evolving business needs primarily through e-capabilities (e.g., without coding) was presented by Lehman [6] in the mid 80's. He reached these conclusions while investigating software productivity at IBM. His analysis, confirmed later by the actual evolution of these ***S-Type*** systems, shows that all these systems experienced **"economic death"** caused primarily by increased software complexity. *Incrementally, it takes longer to deliver fewer features at an increased cost,* making these systems increasingly irrelevant to the business.

The severity of the situation is further compounded by the fact that most features are *developed based on incomplete and incorrect requirements,* reducing further the effectiveness of the solution (e.g., timeliness, cost, robustness). Corporations are littered with numerous *"frozen" legacy systems* that still perform productive functions but can no longer satisfy cost-effectively new business needs**.** This results in the periodic introduction of *additional* systems that target these new needs.

Figure 7 shows that evolving business needs were satisfied through new main-frame (G**3**X), mini-computer (G**4**X) and web based (G**5**X, G**6**X) functionality to complement the original "frozen" legacy systems (G**1**X).

Figure 7: Architecture Evolution



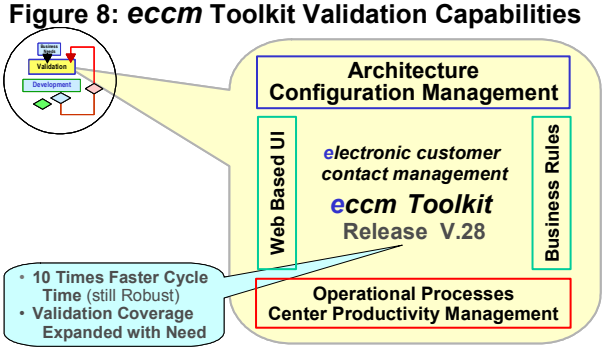
The track record of these multi-system development programs proved to be consistently unpredictable and expensive (e.g., with limited or no ROI). Over 49% of the current feature backlog (e.g., 869/1767 items) requires *concurrent implementation of four or more systems*. Over 24% of the current feature backlog impacts seven to nine systems.

4. *eccm* Requirements Validation *Capabilities*

The ***eccm*** toolkit supports requirements validation capabilities in six key areas related to the order processing center: architecture, configuration management (CM), web based user interface (UI), business rules (BRs), operational processes and center productivity management.

Figure 8 identifies these *key validation areas*. The e-capabilities of each area were expanded in conjunction with the demand for changes (e.g., a new capability was usually implemented if and only if we experienced requirements-churning in a particular area).

Figure 8: *eccm* Toolkit Validation Capabilities



eccm provides an environment in which users can experiment with the solution and introduce improvements through low cost and quick design changes. *There is no penalty for mistakes.*

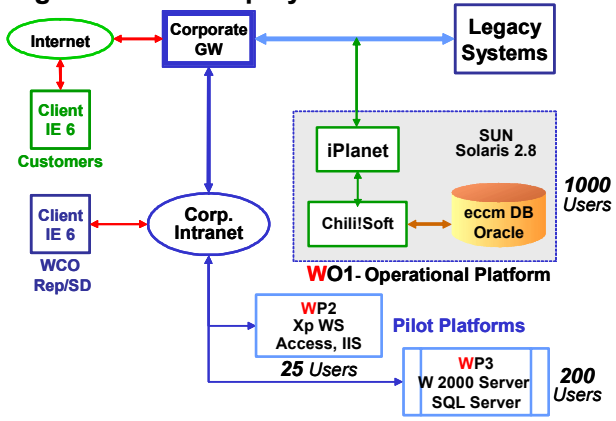
Nordland [9] highlights this gap between verification activities strictly associated with *the baselined requirements* versus *validation of the solution* for the customer's real business needs.
The rest of this paper provides a sample of these validation capabilities in each area. Additional details on the ***eccm*** toolkit's capabilities are provided in previous publications [7, 8].

4.1. Architecture

The architecture provides the environment to validate the solution's *operational robustness* and the specified benefits. A typical two-hour software outage in a 100-person center results in a productivity loss of at least 2 Rep-years (this estimate does not include the cost of government penalties nor the impact on future orders).

Figure 9 shows the use of two pilot platforms for both requirements and solution-benefits validation. Using the WP2 pilot platform, a team of up to 25 Reps is able to process orders and to validate both their individual productivity and the effectiveness of their interactions with outside organizations.

Figure 9: Wide Deployment Architecture



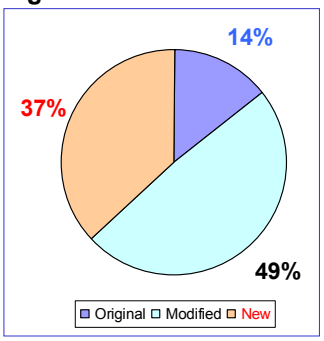
The second pilot platform (WP3) allows up to 200 users to validate center level productivity objectives including the sharing of resources across teams. The solution is deployed into full production on the operational platform WO1 only after the ROI of the new release(s) is guaranteed.

4.2. Business Rules (BRs)

Business rules validation turned out to be a surprisingly difficult challenge. Initially BRs were provided to the development team in a textual description form. Each rule defines a set of field relationships, dependencies and/or constraints. Once the BRs were implemented in software we discovered that no *previously completed* order could pass verification. This was due to incorrect and missing information in the BR descriptions.

In **Figure 10** we demonstrate this point in regard to a high volume, complex 137-field service-order. The BRs were being maintained by a **team of competent experts** and were *considered accurate and complete* (the rules were posted on the company web site with the related order form). It took *five major* iterations to produce a set of validated BRs. *Only **14%** of the BRs were correct; **49%** of the rules had to be modified and, **37%** of new rules had to be added as part of the validation process.*

Figure 10: BR Validation



The process has proven that BRs ***cannot*** *be validated* effectively through standard reviews of textual descriptions.
We have seen three other cases where legacy system developers *were forced to disable* most of the business rules in order to maintain customer electronic order flow. The acceptance of these incomplete orders resulted in high fallout; the solution's FLT could never exceed 30%-35% even after numerous releases.
The ***eccm*** toolkit currently can validate BRs of a complex order within 2-4 hours compared with the previous interval of 4-5 weeks.

4.3. Center Productivity Management

The Center is responsible for manually processing the order fallout. **Figure 11** shows the real time monitoring report that guides each Rep to process the **next most urgent transaction** including work (orders, order supplements), tracking (clarifies and escalations) and administrative status updates.

Figure 11: Real Time Process Monitoring

1. Orders (Rep View)

Process	Rep View	Admin View	Full View
Assign	0	0	2
Work	5 *	0	11 *
Status	2	0	5 *
Track	0	0	1
Done	1	0	1
Total	8	0	20

This guarantees that a team of Reps operates in an optimal mode of transaction processing driven by *a rule-based* set of priorities. These rules can be adjusted without coding in line with the evolving business objectives. This feature has further improved center productivity because it eliminated the need to assign/reassign transactions to a Rep for enhanced accountability and tracking (including the need for reassignments when Reps are overwhelmed or on vacation).
Management reports are provided to track orders, clarifies, escalations, user sessions, business rules and process definitions.

Similarly, the toolkit supports throughput, cycle time and quality reports at the user, team and the center levels. The application currently supports over 200 *validated* reports.

Based on the expertise we acquired, an average time to create a new report is about four hours; iterative validation with the users results in new reports ready to be deployed in the field within 2-3 calendar days. Getting feedback from key users is the gating factor for a new report deployment.

4.4. Web Based User Interface

Users access the ***eccm*** validation environment through industry standard browsers (e.g., IE 6). Order management training of a center Rep takes over **6 months**. User interface (U/I) validation had three human-centric success criteria [7]:

1) *User training should take a day* or less including hands on use of the software capabilities.

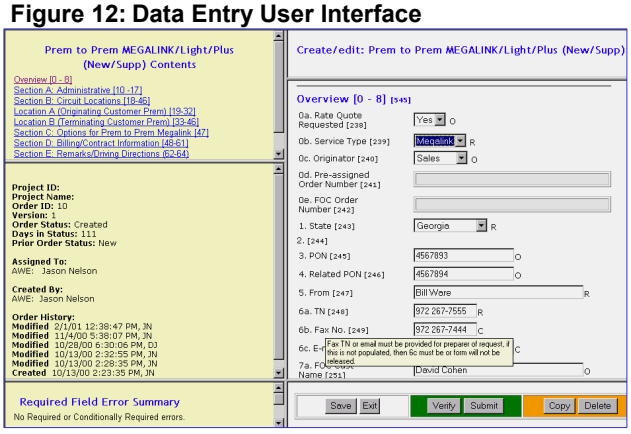
2) The application must support the *baselining of the Rep-capabilities* through measurements collected during order processing using a standard set of test scenarios (it qualifies the Rep's performance using a training database before "going live").

3) New *computer-literate users* must be able to use the application *without training*, by exposing them to a brief introduction presentation (e.g., 15 minutes).

All three objectives have been met.

Figure 12 shows the standard user interface used to create a new order. The *BR knowledge base* was turned into an effective "help" capability using the standard browser capability (holding the mouse pointer over a field provides guidance). This was the most important feature of the user interface, supporting an effective order creation/correction capability with limited/no training.

Figure 12: Data Entry User Interface



The system maintains a **complete** trace of *all order* content and status changes; each trace record identifies the initiator of the action as well as the date and a time stamp.

4.5. Configuration Management (CM)

The following are the key ***eccm*** toolkit CM features (systems engineers use these features to validate requirements without any software development):

1. Create a new order type (or modify an existing one) and the related forms/BRs. A new product can be made available to customers within a week; this is a competitive advantage since in the past it took 4-6 months to introduce a new product.

2. Partition an order form into meaningful logical parts with effective **navigation-links** (see Figure 12). The toolkit supports a powerful order-form editor (e.g., moves fields with the related business rules from one logical section to another, modifies headers, modifies business rules, and automatically updates the navigation links to reflect structural changes, etc.). This capability was implemented to speed up new product introductions or make prompt product updates without the traditional "release planning" process. The implementation uses the "section title" concept of a word processor. Once the new structure is saved, the navigation links are automatically updated to reflect the new structure.

3. Classify data fields as ***R**equired* (a user must supply a Required field with an appropriate entry when creating/editing the associated form), ***C**onditionally **R**equired* (the field value is determined through a Boolean expression of other field values) or ***O**ptional* (field names can be modified as needed). Each data field can be assigned integrity constraints and/or BRs.

4. Modify a variety of system parameters (e.g., time out parameter of an inactive session - 30 minutes; time interval used in active/inactive order reports - 30 days).

5. The toolkit provides system administrators capabilities to maintain customer accounts, teams, team administrators, users, user sessions, user passwords, database backups and software release updates (see **Figure 13**). Throughput and quality reports are provided to administrators to track all CM activities.

Figure 13: Key CM Capabilities

6. The toolkit supports an **UNDO** status capability. The user interface warns the user before an order is cancelled and the action is taken only *with direct user concurrence*. If the user makes a mistake in spite of the warning, the UNDO capability facilitates restoration of the database to the previous state while still tracking the UNDO action.

4.6. Operational Processes

The operational processes have a dominant influence on the center's productivity. They define the organization's internal and external interfaces. Each one of these interfaces represents a handoff between organizations and/or people, and thus a *potential cause for delay and/or confusion*.

For example, a new order is created by a customer. It is a data entry function that requires a significant amount of knowledge about the circuit to be provisioned. The data entry function can be suspended at any point in time and resumed later using the saved order data. The system keeps track of *all* changes in the life of the order. This trace database is an important element in the production of the management reports. At any point in time the user may verify the accuracy and completeness of the order according to the business rules specified for this particular product.

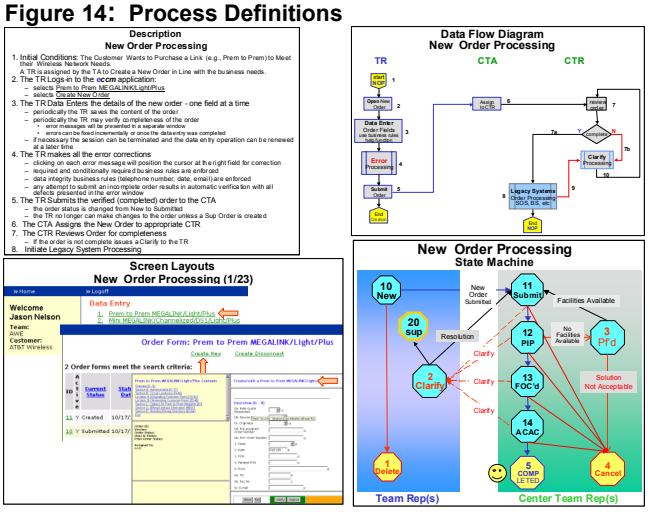
Once the order has been completed (some products include over 120 fields and almost 200 business rules) the customer may submit the order for center processing. **Verify** processing was integrated into the submit function; *an order cannot be submitted by the customer unless it has been verified for completeness and accuracy.*

The *submission of an order* automatically transfers responsibility from the customer team to the center team, and specifically to the center team administrator (CTA). The *CTA assigns* the new order to a particular Rep in his team according to expertise, availability, workload and/or urgency. Both activities are tracked by the system. The Rep evaluates the order and if it is complete, legacy system orders are issued.

Figure 14 shows the definitions of "create new order" process. Each operational process was defined in four standard formats: *step by step text*, *dataflow diagram*, *state machine* and *step by step U/I screen dumps* while executing the required tasks.

No *single description format* was effective in training large number of Reps; therefore, each process was specified in **all four** description formats.

Figure 14: Process Definitions



Additional high priority was given to process definitions because they are required to ***verify completeness*** of the functional requirements. A typical project misses about **30-40%** of the needed requirement specifications if the baselined operations processes are not defined, documented and understood.

Initially all process changes required software development that negatively impacted operational productivity and robustness. Currently, **table-driven state machines** control all order management processes. Most process changes (e.g., create new states, or make changes in the state transition matrix) can be implemented through CM updates.

5. Conclusions

eccm has proven to be an effective toolkit to:

1. **Eliminate software bottlenecks in the delivery of business solutions**, *shifting the challenge from software development to change management.*

2. **Improve delivery of *relevant* and *complete* solutions to center users.** The total cost of legacy-system software releases has been reduced by at least an order of magnitude compared to similar past projects because we deliver complete and effective solutions the first time, eliminating the massive rework of the past. We also improved the timeliness of each solution by a factor of five or more; past cycle-times of 2-3 years have been replaced with cycle-times of 4-6 months.

3. **Expand the life cycle of existing legacy systems** by eliminating the unnecessary churning of software features while concurrently improving their software ROI (e.g., using the ***eccm*** toolkit to validate business rules). New legacy system releases, using past investment levels deliver on time solutions with a 40% quality improvement (defect density).

4. **Enable the transitioning of the software industry from a *requirements-driven* to a *ROI-driven* paradigm.** The validation toolkit provides development organizations a proven and predictable methodology to deliver business-relevant solutions vs. *software functionality that does not benefit operations.*

6. References

[1] B.W. Boehm, "Software Management," *IEEE Computer*, September 2000, pp. 94-97.

[2] B.W. Boehm, and V. Basili "Gaining Intellectual Control of Software Development," *IEEE Computer*, May 2000, pp. 27-33.

[3] B. W. Boehm, "High Dependability Computing in a Competitive World," *IEEE Proceedings of 26th SEW IEEE/NASA*, November 2001.

[4] P. J. Denning, *"The Invisible Future, When IT Becomes a Profession,"* pp. 295-325, McGraw Hill 2002.

[5] D.E. Avison and G. Fitzgerald, "Where Now for Development Methodologies," *Communications of ACM*, Vol. 46, Number 1, January 2003, pp. 79-82.

[6] M. Lehman, *"Program Evolution: Processes of Software Change,"* Acad. Press, London, 1985.

[7] D. Cohen, G. Larson, Bill Ware, "Improving Software Investments Through Requirements Validation," *IEEE Proceedings of 26th SEW IEEE/NASA*, November 2001, pp. 106-114.

[8] D. Cohen, G. Larson, Bill Ware, "Delivering E-Type Solutions through Requirements Validation," *The 6th World Multiconference on Systematics, Cybernetics and Informatics Proceedings IIIS and IFSR*, July 14-18, 2002, pp. 224-229.

[9] O. Nordland, "V&V – Veridation or Valification", *The 6th World Multiconference on Systematics, Cybernetics and Informatics Proceedings IIIS and IFSR*, July 14-18, 2002, pp. 261-266.

[10] P.J. Denning and R. Duncan, "The Missing Customer," *Communications of ACM*, Vol. 46, Number 3, March 2003, pp. 19-24.

[11] B.W. Boehm, and L.G. Huang "Value-Based Software Engineering: A Case Study," *IEEE Computer*, March 2003, pages 33-41.