A DECISION MODEL FOR MANAGING SOFTWARE DEVELOPMENT PROJECTS

Thang N. Nguyen
California State University Long Beach
thangn@csulb.edu

ABSTRACT

There are four issues, rather fundamental, with current software development management practices, (1) unmatched structures between software developing models and organizations driving them, (2) process-centric versus interprocess-centric, (3) separation of quality and metrics modeling from process modeling and (4) little attention paid to the building of automated, practical, relatively simple and effective decision models. The four issues hinder the effective management of, and decision-making on, quality software development process and products delivery by practitioners. This paper suggests a decision model for managing software development projects that is highly quality-assured. The model is based on four concepts: mappability (what and why) addressing the first issue, accountability (who) addressing the first issue, interoperability (how) addressing the second issue, controllability (where) addressing the third issue, for decision making (when) addressing the fourth issue. The decision-making is based on a set of indicators on deliverables between processes at the completion of tasks. The indicators reflect criteria that are measurable and are associated with severity status. The quality is assured based on a set of four attributes: completeness, correctness, consistency and compliance (C3C). The decision model allows efficient decision making to be made at appropriate level of management. A prototypical web-based implementation is presented and discussed.

Submitted to
Journal of Information and Management

Previous refereed conference version appeared in
Proceedings of IEEE International Conference on System, Man and Cybernetics, Tokyo, Japan 1999
as

A Quality-Assured, Hierarchical Model for Solution Development Management
A DECISION MODEL FOR MANAGING SOFTWARE DEVELOPMENT PROJECTS

Thang N. Nguyen
California State University Long Beach
tnnguyen@csulb.edu

1. ISSUES

There has been a myriad of publications and still growing research effort addressing many different aspects of software development project management (Pressman, 2003). There has been also many models (e.g. CMM), methods (e.g. WBS), techniques and tools (e.g. Microsoft Project) supporting the practices (Purba et al. 2000, State Services Commission 2001) of all phases of the different software development life cycles for achieving the conflicting objectives of fast delivery and lower costs of higher quality products. These practices cover project planning, software requirements, high-level design, low-level design, implementation and testing (Williams, Clay E. 1999), estimation and scheduling, software metrics and measurement, risk management (Boehm, 1991, Fairley 1994), and software costs (Chulani et al., 2000), to name a few.

The models (Microsoft, 2003), methods and techniques (McConnel, 1998), practices and tools (Starwright, 2002) can be very broad and generic or can be focused and specific, depending on the perspective or motives that the researchers and practitioners are looking at the software development/software project management. Some authors argue that planning is the key (Srinivasan, 2003). Others place the emphasis on people. A large group advocates that the focus should be on process (McAndrew 1993), and that the process needs be improved continuously while practitioners are busy building software project management tools. As a result we have a wide range of products that are becoming very sophisticated, complex, detailed and expensive.

Overall, there are however four issues, rather fundamental, with current software development management practices, as argued below:

- **Unmatched structures between software developing models and organizations driving them**: software development models are linear, following the traditional sequence of Requirements-Design-Implementation phases (McConnell, 1996) while the overall organization that manages the development effort is hierarchical. These two unmatched structures obscure different granularity levels in task management, introduce conflict of authority and policy, and tolerate inappropriate decision-making.

- **Process-centric versus interprocess-centric**: the current software development’s focus is on the processes supporting the development, rather than on the interfaces (i.e. interprocess) between different phases or subphases (if any) of the development cycle. This results in mismatches and gap between the sending organization units and the receiving organization units. Mismatches occur when the receivers get more (less) deliverables than what they actually expect. Gaps occur when the receivers must perform some additional tasks before the deliverables can be useful. Improvement for better
product delivery is also process-centric and frequently piecemeal due to resource constraints on implementing changes. Global and continuous improvement of the entire organization process would be too expensive and time-consuming, an issue already widely recognized (Park et al., 1996)

- **Separation of quality and metrics modeling from process modeling:** in practice, quality criteria of products and of processes supporting them, as well as metrics, are formulated separately from development process modeling; therefore, consistency and practical benefits are in fact difficult to obtain (Barbacci et al., 1995), and

- **Little attention paid to the building of automated, practical, relatively simple and effective decision models:** in fact, automated models for managing software development projects exist but mostly are cost-driven models or time-driven models, rarely have we found an online decision model with complete rolled-up status of, and drilled-down details from the offending tasks or incomplete deliveries, for timely and effective decision making by all responsible parties. Furthermore, the management of the development effort is commonly placed at the project level with project managers/leaders commonly selected from technical ranks (McCarthy, 1995) rather than at the business level or at a higher level of management (although participation from non-R&D organizations such as marketing, sales, support and services is solicited). Therefore, predefined business goals for the project are at times forgotten, shifted, not observed or tracked. This is partially because the development may gradually be detached from business scope as a result of the primary ownership of development generally assigned to Research &Development (R&D) organization.

The four issues mentioned above hinder the effective management of, and decision-making on, quality software development process and products delivery by practitioners (Nguyen, 1999). Furthermore, the nature, scope, size, complexity of software development projects as well as resources (talent, funding, technology, and others) are quite different and vary largely, from short-term Web-based applications to long-term e-business integration within an enterprise or across enterprises. It is often required that different decision models are needed for different projects.

This paper suggests a decision model for managing software development projects that is highly quality-assured. In section 2, we formulate our model based on four concepts: mappability (what and why), accountability (who), interoperability (how), controllability (where) for decision making (when). The decision-making is based on a set of indicators on deliverables between processes at the completion of tasks. The indicators reflect criteria that are measurable and are associated with severity status. The quality is assured based on a set of four attributes: completeness, correctness, consistency and compliance (C3C). The decision model allows efficient decision making to be made at appropriate level of management. A prototypical web-based implementation is presented and discussed in section 3. Benefits and values of the model are summarized in section 4.

2. **A DECISION MODEL FOR EFFECTIVE MANAGEMENT OF SOFTWARE DEVELOPMENT PROJECTS**
Our software development project, for illustration purpose, consists of 4 typical stages or phases, following a conventional model (note that it could have more phases, if so desired), namely (phase 0) **project definition (feasibility)**, (phase 1) **requirements**, (phase 2) **design**, and (phase 3) **construction (implementation)**. This simplified phase model serves as the basis for discussing the following essential concepts corresponding to the five W’s and the H of the decision model to be constructed (i.e. who is doing what, why, how, where and when): (1) the mappability between the development process and the organization driving it (the what and why) by the use of goal-directed indicators, (2) the accountability (the who) to include the roles and responsibilities of all parties involved in the software development project, (3) the interoperability (the how) based on the interprocess concept and the quality assurance (how well) with basic quality concepts (correctness, completeness, consistency and compliance), (4) the controllability at the interprocess interfaces or gates (the where) to eliminate mismatches and gaps between processes, and (5) finally the effective decision making (the when) using keys to indicate severity level (Red - critical, Yellow - warning or Green – normal) associated with the completion of a task or a deliverable by a responsible party. The model is elaborated below.

**The what and the why: Hierarchical mapping between software development process and organization driving it**

![Hierarchical mapping between process and organization](image)

*Figure 1: Hierarchical mapping between process and organization*

The issue of unmatched structures between linear development life cycle process and hierarchical organization driving it suggests the idea of restructuring the flat nature of the development process to map to the hierarchical nature of the organizations driving and
supporting it (mappability). Figure 1 shows the structure of such mapping. The development process consists of 4 conventional phases as shown (left side – may be any number of phases). The organization is described as a hierarchical structure of 4 levels of management (right side - may be any number of levels of management). The mapping is based on a set of goal-directed indicators (in the middle) similar to those obtained by the goal-driven indicator technique discussed in Park (Park et al. 1996).

<table>
<thead>
<tr>
<th>Phase 0 /Tasks</th>
<th>Indicator texts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Definition</td>
<td>Project definition in place, on time &amp; under budget – executive concern</td>
</tr>
<tr>
<td>#1</td>
<td>Business case presented – project/first-line mgmt/non-mgmt personnel</td>
</tr>
<tr>
<td>#2</td>
<td>Project scope and objective defined – project/first-line …</td>
</tr>
<tr>
<td>#3</td>
<td>Customer Satisfaction objectives defined – project/first-line …</td>
</tr>
<tr>
<td>#4</td>
<td>Project costs and revenues estimated …</td>
</tr>
<tr>
<td>#5</td>
<td>Risk assessment complete and success factors identified …</td>
</tr>
<tr>
<td>#4</td>
<td>Quality objectives defined …</td>
</tr>
<tr>
<td>#5</td>
<td>Estimates and Schedules completed …</td>
</tr>
<tr>
<td>#6</td>
<td>Pricing model complete – project/first-line mgmt/non-mgmt personnel</td>
</tr>
<tr>
<td>#7</td>
<td>….</td>
</tr>
<tr>
<td>Requirements</td>
<td>Requirements and features identified and in place – executive concern</td>
</tr>
<tr>
<td>#1</td>
<td>User requirements collected and complete – project/first-line …</td>
</tr>
<tr>
<td>#2</td>
<td>…</td>
</tr>
<tr>
<td>Design</td>
<td>Design of features complete and on time – executive concern</td>
</tr>
<tr>
<td>#1</td>
<td>….</td>
</tr>
<tr>
<td>#2</td>
<td>….</td>
</tr>
<tr>
<td>Construction</td>
<td>Builds, tests and product launch – executive concern</td>
</tr>
<tr>
<td>#1</td>
<td>….</td>
</tr>
</tbody>
</table>

*Figure 2: Excerpts from Project Indicators*

The mapping is described as follows. Each completion of a task, a phase, a deliverable of a task or phase within a development process is associated with a goal-directed indicator. Figure 2 shows a simplified phase 0 tasks (Project Definition) and the indicator associated with each task. An executive would be interested to know if the Project Definition Phase 0 is in place, on time and under budget. This indicator and those corresponding to other subsequent phases (phases 1, 2 and 3, partially shown, marked executive concern in bold) will constitute the indicator set associated with the executive sponsoring the project.

Similarly, a project manager would be interest in monitoring the completion of all tasks defined within each of the phases as the project is progressing (e.g. Phase 0, tasks numbered from 1 to 7). The details of all these 7 tasks are of the project manager’s concern. The Project Definition phase (Phase 0) may consist of business case, objectives, costs, pricing model, and plans for requirements gathering, architecting, designing and realizing them. The plans may include a marketing plan, sales plan, support plan, service plan, training plan, quality assurance plan and,
of course, R&D plan. In these plans, the focus is on the schedules, estimates of resources and tasks or activities to be performed. These items are responsible by Project Management but also involve all participating organization units contributing to the success of the phase. Some of the tasks may be assigned to a particular first-line manager or non-management personnel such as business analyst, developer, tester, builder or technical documentation personnel. The indicators associated with all tasks owned by project management constitute the set of indicators for project management level. The subsets of indicators corresponding to the tasks assigned to a particular first-line manager constitute the set of indicators for that particular first-line management. And a particular indicator may be associated with a particular tasks or deliverable responsible by a particular non-management personnel. Figure 2 in fact shows a re-arrangement or re-distribution of phase indicators to different levels of management.

For the subsequent phases, the collection of identified functional and feature requirements making up the Requirements Specifications phase (Phase 1 – in our example model) corresponds to another set of defined indicators, again with sufficient participation and collaboration of all participating organization units. The Design phase (Phase 2) is commonly assigned to an R&D organization unit (primarily to First-line Management). This is the phase where the high-level design and low-level design take place. Features and functions codes, software builds and tests result in the form of one or more CDROM images or libraries (Phase 3 – Construction, also commonly responsible by non-management personnel and its first-line management) with their own indicators. The codes after going through testing, certification (quality-assured) and build constitute the final products or releases and are ready for Field Test (FT), First Customer Ship (FCS), or Early Support program (ESP), and finally General Availability (GA).

Thus the mappability results in a hierarchical structure of tightly-coupled indicators corresponding, on the one hand, to every and all tasks completion and deliverables of the four phases, and on the other hand, to different levels of decision-making management of the responsible organization.

**The who: Accountability**

Essential elements (Ackoff, 1990) for effective strategic, tactical and operational planning are accurate scope of work with well-defined mission (i.e. the business we are in), objectives (i.e. target to reach within the framework), strategies (i.e. how to achieve objectives and/or how to compete), and tactics (i.e. tasks to achieve in short-term). But no matter how well sound the planning is, if accountability is not enforced, no appropriate and timely decisions can be reached. The model allows accountability to be exercised.

In fact, the mapping scheme described earlier implies accountability. During the formulation of the mapping, we assign one or more indicators to a specific management level or to a particular practitioner. For example, the Project Definition phase indicators in Fig. 2 have a set of indicators (e.g. #1 to #7) defined such that each is owned by non-management personnel, a subset of indicators owned by first-line management, the whole phase owned by project management, while the complete set of indicators is owned by an executive. All indicators are to be monitored, evaluated, reported and rolled up by every lower level of management from its
own perspective and scope of accountability. The complete set of indicators with assigned accountability is expected to encompass all aspects of the development and delivery cycle.

The how: Interoperability

Common practices in development have (1) historically placed the emphasis on process and its sub-processes (the issue of process-centric development) for task management and change management, and (2) considered project-level management as dominant (the issue of level of project management and decision making), therefore played down the continuing influence of higher-level management as previously argued in section 1. The first understates the importance of operational interface between processes (called inter-process) and the second inadvertently confines the development effort to a technical scope rather than a business scope and tolerates business decisions being made by technical practitioners.

By shifting the emphasis from process to inter-process for interoperability (Fig. 3) and elevating the management of deliverables, process, resources and environment beyond and above project management level, the organization would be able to manage the development and delivery from an overall view encompassing business, marketing, sales, customer satisfaction views, as well as technical and support views.

Interoperability is defined at the interface between two organization units: one unit is the sender and the other is the receiver. Closer examination of the interoperability concept reveals that interoperability at all levels of granularity in any organization yields a practical way of detecting mismatches and gaps. The sender may give a deliverable in a format that does not totally match the expectation of the receiver (mismatch). In addition, the receiver may receive an item that requires more work before it can be accepted for further processing (gap). Both mismatch and gap require rework or extra work. Keys to the successful interoperability are the deliverables and the criteria governing them. To eliminate mismatch and gap in deliverables for mutual acceptance between sender and receiver, the interface between processes (or sub-processes) acts as control point for delivery validation and acceptance based on exit criteria from one process and entry criteria of the next, called keys.

The indicators are associated with keys to define the status of the task completion or deliverable at the boundaries between processes (i.e. gates). The keys are measurable. For example, if an indicator for the entire project status is “Is the project on time and under budget”, then the following may be defined as keys to determine severity of its indicator status (red, yellow, green):

---

Figure 3: Process and interprocess
During the project life, if any date of the planned schedule is slipped, the project is not on time. If any of the project items turns out to actually cost more than estimated, the project is cost-overrun.

- If the lateness is more than 2 weeks and/or the cost is more than $20,000, then the project is on Red status (critical).
- If lateness is less than 2 weeks and cost is less than $20,000, then the status is yellow (warning) else
- it is green (normal).

Figure 4 shows an excerpt example of the deliverable dates between two authorities/phases. Exit criteria must be verified by sender (sending authority/organization) and deliverables validated by receiver (receiving authority/organization) based on entry criteria. Sign-off may be required at this interface. An example is shown below.

- If the difference between planned delivery date and actual delivery date (or between actual delivery date and actual receiving date) is above a certain duration (e.g. one week), a red indicator can be flagged.
- If it is less than one week, it may be flagged as yellow.
- If the dates are same, then the delivery is on time (Green). Reason for lateness may be documented.

The general business rule is that the delivery may be checked against certain criteria agreeable a priori by both authorities. If more than one criterion is not satisfied, the delivery may be flagged as red. If only one criterion is met then the indicator may be termed as yellow. Green is flagged when not lateness, no mismatches and no gaps are encountered.

<table>
<thead>
<tr>
<th>Sending authority</th>
<th>Receiving authority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Planned delivery date</td>
<td>Planned receiving date</td>
</tr>
<tr>
<td>Actual delivery date</td>
<td>Actual receiving date</td>
</tr>
<tr>
<td>Exit criteria met</td>
<td>yes no</td>
</tr>
<tr>
<td>Entry criteria met</td>
<td>yes no</td>
</tr>
</tbody>
</table>

Figure 4: Example of Deliverables status (interoperability)

The where: Controllability at the interface (interprocess) to ensure quality assurance

The issue of separating quality modeling from process modeling, as seen in the past, suggests the need for a single, integrated point of departure from which process, deliverables, resources and environment are examined together with their quality attributes and metrics in support of controllability. The indicators allow metrics to be defined, therefore measurable.

A taxonomy of quality attributes discussed in Barbacci et al. (1995) in the areas of performance, dependability, security and safety, can be combined with well-known quality standards (American National Standards, 1994) and guidelines as published by various quality assurance/quality control groups and organizations. These attributes offer an appropriate set of
basic measurements when they are viewed at the inter-process level, namely completeness, correctness, consistency and compliance that govern the deliverables. In other words, the hierarchical sets of indicators are examined to make sure each task completion and each deliverable in accordance to those quality attributes guidelines. Quality metrics are thus associated with each and every indicator, whether it is an independent indicator (at the non-management personnel level) or aggregate indicator (at the higher level of organization and management). The model is therefore quality-assured based on C³C.

Throughout our development and delivery process, primary control is placed at the interface between two subsequent phases. (The control within each process or sub-process supporting the corresponding phase is the responsibility of the corresponding owner). Controllability associated with the phase indicators allows decision-making to be easily exercised whether the decisions are strategic, tactical or operational.

Both interoperability and controllability are performed at the inter-process level. Interestingly, this in turn allows a direct linkage between process attributes and quality attributes of deliverables for management consideration at all levels rather than just at the project level as commonly seen in practice.

The when: Effective decision-making

As we have indicated, to aid in decision making, at any stage of development and delivery, status is reported by the set of indicators describing it. The lower indicators rollup to support the next higher level of indicators for decision making. Decisions at any phase are based on deliverables’ status at the end of the phase. This allows for the next phase’s activities to be initiated. Information gathering during the process is readily available so that decisions makers can apply different decision schemes based on situations and functions.

As a general rule, every organization unit and every level of management are involved in the decision-making process, each in its own scope of responsibility, level of authority and roles (primary or supporting). A decision scheme using the indicators that involves three core tasks as described by Rouse (Hopple, 1986) can be readily applicable: (1) situation assessment can be used in conjunction with the set of indicator statuses, (2) planning and commitment and (3) execution and monitoring may be incorporated in deliverables reporting. Thus, the model allows other methodologies on decision processes (Hopple, 1986) to be implemented as appropriate.

3. A WEB-BASED EXAMPLE IMPLEMENTATION USING THE DECISION MODEL

A prototype implementation described below will serve as proof of concept for the decision model. In this simplified prototype, we use the 4-phase approach (1) project definition (feasibility), (2) requirements, (3) design, and (4) implementation (Figure 3), and the organization consists of three layers of decision making: the non-management personnel at the lowest layer, the project manager at the mid-layer, and the executive sponsoring the project at the highest layer.
A major deliverable is the Phase Definition Report (Figure 5b). The indicator texts are the subheadings (tasks) called work items of the report. A deliverable is the detailed write-up under each subheading (e.g. business case, purpose, etc.). The evaluation rule is that if the section under a subheading is well prepared and well written, then the indicator is marked as Green. If missing, it will be Red. Some incomplete description will result in a Yellow.

**Roll-up status and drill-down details**

When a responsible personnel prepares the Phase 0 Definition document (report) or artifact, s/he submits to higher authority for review. The immediately higher authority (a project manager, in this prototype) in turn evaluates the document content (Figure 6b) to report a Green, Yellow or Red status (Figure 6a) depending upon the status of the write-ups (deliverables) and the quality (Correctness, completeness, consistency and Compliance) of the items under review. Examination of the report by a project manager (in this prototype) concludes that there are four work items that are approved (Green indicators), and one item (Project Estimates and Schedules) is not (Red indicator).

The rules for mark up indicators are as follows. A green in Phase 0 requires that all Phase 0 Work Items are green. A Yellow requires that there is at a Yellow in a work items without any Red) in any work items will result in a Yellow in Phase 0. Any Red (regardless of others) will result in a Red indicator in Phase 0. This would make the whole Phase 0 Red, therefore it will be automatically rolled up to report as Red in the Executive Summary report (Figure 6). Other
phases (Phases 1, 2 and 3 – not shown) will work the same way, i.e. having the same roll-up status capability. Any phase (0, 1, 2 or 3) that is Red will result in a Red the project status.

Inversely, the above roll-up reporting scheme allows detailed drilled-down operations to be performed. The responsible executive, after his/her login, observes that a particular project (e.g. Online Project Tracking) showing an overall status of R (Red) in Phase 0 of its project. The executive can drill down to see the status of the project in any phases by clicking the appropriate links (e.g. the gray button associated (Link to Phase Details) with the phase 0 in Figure 6 to obtain the Phase 0 work items report (Figure 5a). Continuing the drill-down review by clicking the gray button (Link to Actual Documents), the executive is presented with the actual document showing the reason for the R status: missing Project Estimates and Schedules.

The prototypical application was basically designed to implement the four concepts discussed in section 2: mappability (what and why), accountability (who), interoperability (how) and controllability (where) for decision-making (when) for all levels of management with C³C. It also provides the capability of on-line submission of reports on deliverables and/or links to other deliverables such as program source code libraries, test libraries, production libraries and others reports with document links and data base links to appropriate repositories.

Drill-down details:

An executive can drill down the details on why a particular phase of the project s/he is sponsoring has a Red or Yellow status by clicking on the gray button (link to phase details). Figure 5a shows Phase 0 Work Items in which the Red status was marked due to Missing Project Items in which no Project estimates is found. (Figure 5b)
This, a user, depending upon his/her position in the organization is presented with a view of only projects s/he is responsible for. An executive would see all projects that s/he is sponsoring. A project manager will see only his/her project. The technical manager or technical leader oversees the development of his/her project. The higher the position in the organization, the broader and the coarser the associated view.

Since the project is in Red status, the executive, at his/her discretion may choose to override it to allow the development effort to continue. The executive must, however, document why s/he selects to change the status from Red to Yellow or Green. By changing the radio button on the right side of the Executive Summary window (Figure 6), the executive is presented with a dialog (not shown) where his/her justification for status change is permanently recorded and undeletable in the application. The justification is automatically sent to all parties involved in the relevant work item/effort. The executive may also request a list of all projects owned by him or her, which have a particular status such as all non-Green projects. Project Management and first-line management can also override their associated indicators and keys within their scope of responsibility.

Besides the roll-up status, drill-down details, executive override and management override, the prototype offers other capabilities. A complete list of capabilities provided by the prototype is summarized below:

- on-line submission of review reports with complete editing capability: conceptualization documents such as requirements definitions, design specifications can be prepared, placed online and linked to
- on-line approval and control of process by virtual meetings: physical meetings for managing the projects can be minimized as the online application allows timely reporting and action to be taken
- access and action control by appropriate management level
- automatic roll-up status and drill-down details as shown in details in the previous sections
- deliverables monitoring, quality assurance and quality control implementation (completeness, correctness, consistency and compliance: C^3C): this may be used as a basic attributes for defining higher and more abstract attributes such as availability, reliability and the like
- status override function with required (undeletable) justification while avoiding conflict of authority
- reports on outstanding items (status, schedules, issues, actions): the reports can be automatically published with and without notifications to all personnel parties involved

4. DECISION MODEL MAKES A DIFFERENCE - CONCLUDING REMARKS

We have discussed and implemented as prototype the functionalities of our hierarchical, quality-assured decision model. The pilot implementation has revealed many benefits. In fact, with a web-based application, all reports submitted online with pre-formatted forms can be easily accessed by authorized personnel at any time from anywhere. This minimizes the need to hold physical meetings (hence virtual meetings) between the parties involved unless it is really
necessary. Each responsible party will work on its own time to report progress and status on its own tasks.

Since the Red, Yellow and Green status reported at the bottom level of task hierarchy is rolled up automatically to the highest level of management: executive, the executives may make strategic and tactical decisions or override the status with proper justifications recorded and undeletable. The overriding notifications are automatically forwarded to responsible parties involved. The executives may drill down the task hierarchy to examine any details for proper action items. From a quality assurance and quality control perspective, the use of four principles: completeness, correctness, consistency and compliance at any management and technical levels help identify errors and defects in solution planning, estimation, sizing, scheduling in all phases of development cycle from conceptualization to realization. Different and special reports on outstanding items can be generated and separated by project, owners, and other types of grouping.

In summary, the paper proposes a decision model as solution to the four issues cited in section 1. As described in the sections 2-4, the making of our decision model relies on the mapping between development process and the hierarchical organization supporting it. The key to the mapping is the set of goal-directed indicators targeted at the completion of tasks and deliverables by each task. The creation of these indicators is completely within the control of the organization units in charge.

The decision model described in this article thus offers the software development organization a quality-assured, decision-assisted management scheme with high practicality, simplicity and flexibility via the use of a hierarchy of goal-directed indicators. The indicators have a direct relationship to metrics and quality of process, deliverables, resources supporting them and environment. The model may use quantitative and/or probabilistic metrics described by Florac (Florac et al., 1999). The organization may make use of any existing measurement techniques and metrics associated with them, and any decision making schemes. The metrics serve as keys to severity level for each indicator.

From the software development process point of view, any life cycle model and/or process improvement can be adopted. The model works well with any software process frameworks (e.g. CMM or CMMI) (Chrissis et al., 2003), or processes, simple or complex, traditional or for e-business solution development and delivery. This is because it is conditioned only by appropriate indicators defined by the organization according to its mission, goals and strategies. It will also work well for e-businesses application development, enterprise application integration and others types of application development such as portals, web-based or web-enabled application or e-business integration. Furthermore, the emphasis on inter-process adds greater control at different levels of granularity while allowing continuous process improvement globally and/or locally. From the management organization perspective, any organizational structure can be mapped.

5. REFERENCES


Chiriss, Mary Beth, Mike Conrad and Sandy Shrum. (2003), CMMI: Guidelines for Process Integration and Product Improvement, Addison-Wesley. 2003


