

# *CDS: Reconciling Management Objectives and Operations*

*An Integrated Framework for Managing Product Development*

---

Product Value Management

May 2005

Collaborative Product  
Development Associates, LLC  
222 Grace Church Street  
Port Chester, NY 10573  
(800) 573-4756  
[www.cpd-associates.com](http://www.cpd-associates.com)



## **CPDA: Collaborative Product Development Associates, LLC**

CPDA's Product Lifecycle Management (PLM) research programs target the critical decisions in Product Lifecycle Management challenging Design, Engineering, Manufacturing, and Information Technology managers and executives. CPDA's PLM collaborative research programs provide in-depth analysis of strategies, products, issues, processes, technologies, trends, case studies, and surveys for assessing technology, business goals and objectives, and implementation road maps.

The cohesive suite of collaborative programs clarify and evaluate new capabilities, frameworks standards, and development issues; they highlight the most advanced uses of leading technologies, and they link the technical effort to the realization of business value. The four collaborative research programs include:

- ***Design Creation and Validation:*** A bottom-up view of engineering requirements from the desktop across the enterprise.
- ***Product Definition:*** A top-down view providing a conceptual framework for tightly coupled collaboration across different product development perspectives, bridging Customer Needs, Systems Engineering and Tradeoffs, Design Solutions, and Fulfillment and Manufacturing.
- ***Product Value Management:*** Mapping business process modeling to an IT integration infrastructure in a Federated Enterprise Reference Architecture (FERA)<sup>TM</sup> loosely couples multiple touch points within and between enterprises extending across the supply chain.
- ***PLM Infrastructure:*** Integration and interoperability in complex PLM environments pose substantial hurdles. The rapid transition to cross-enterprise collaboration, at all levels of design and supply, intensifies the pressure on existing, inwardly focused IT architectures to support and enable new modes of doing business.

Collaborative Product Development Associates was formed by the PLM research team of D.H. Brown Associates, Inc. (DHBA).



# *CDS: Reconciling Management Objectives and Operations*

*An Integrated Framework for Managing Product Development*

*Vasco Drecun, PLM Research Director*

## **EXECUTIVE SUMMARY**

CDS (Chrysler Design System) has evolved for several years into a comprehensive framework for managing the process of product development at Daimler Chrysler. This paper summarizes the conclusions of a Feasibility Assessment of CDS as an Integrated Process Framework. The targeted two objectives are:

- An evaluation of the level of integration between both process and metrics models, e.g. the feasibility to represent the entire CDS framework encompassing both models as an integrated framework for process analysis and improvement.
- An assessment of any gaps between metrics and process models, and their impact to the company's ability to fully reconcile governance with activities.

IBM, Intel and PRTM funded the effort, and contributed directly to the content. CPDA (Collaborative Product Development Associates, LLC) directed the project, with valuable contributions from Value Chain Associates, LLC.

## **HIGHLIGHTS**

### **CDS 4.3**

Overall, CDS 4.3 is a well-balanced framework. CDS 4.3 directly reconciles deliverables and their gate exit criteria with the tasks required to produce those deliverables across all activities and phases of development. A critical value, CDS 4.3 directly links management with operations.

As targeted in the project, VCOR and the Value Scape tool identified modest inconsistencies that resulted from the manual maintenance of a large knowledge base. Considering the manual maintenance and the size of the existing system, it was surprising not to find even more problems in the knowledge data base.

The framework could be extended in two significant areas:

- The structure of 4.3 could be mapped to a finer level of decomposition of tasks and I/Os characterizing operational requirements, and specific to the sub-systems' development, such as chassis, body, interior, electrical, or powertrain.

- Enhancements would be required to support critical path analysis, even though CDS 4.3 does map the dependencies between program tasks. These enhancements would bolster its usability for automated reporting and tracking of development programs.

#### **CDS 5.0 TARGETS**

While CDS 5.0 currently represents a “work-in-progress,” it currently defines a framework for metrics-based program management. For that purpose, it incorporates a reporting template for a comprehensive set of metrics to evaluate both effectiveness and efficiency of the development effort. With the transition from a deliverable-based program management to a metrics-based program management, DaimlerChrysler is trying to strengthen accountability and to sharpen the focus of key decision makers on the most critical performance gauges. The transition represents a natural step in the maturation of program management.

*CDS 5.0 should also explicitly identify and manage critical dependencies during the development process, which was one of the strongest capabilities of CDS 4.3. CDS 5.0 would thereby extend the established links between operational engineering and management. By maintaining deliverables along with the associated reconciliation with tasks and I/Os, CDS 5.0 supports an accurate and complete understanding of program risks. Moreover, that effort would also facilitate the transition from deliverable-based to metrics-based reporting.*

Several proposed enhancements to the CDS 5.0 structure may also promote a smooth transition to 5.0 that builds on the demonstrated strengths of CDS 4.3. Most critically, it is highly recommended that the CDS team consider rolling out CDS 5.0 with the tasks and deliverables associated with the process streams explicitly identified and reconciled with their I/O dependencies. The dual structure would significantly augment the metrics reporting approach, which already benefits from the development of comprehensive gate templates. It strengthens the understanding of how different metrics map to the dependencies between activities of different groups in the internal and external value chain.

Reusing knowledge contained in CDS 4.3 and extending the structure of CDS 5.0 to enable both metrics-based reporting and dependencies management would be a sound approach for DaimlerChrysler.

Table 1 on the following page is a proposed road map comprised of three enhancements. Several benefits could be obtained by implementing these as a set of incremental framework releases, each targeting specific benefits.

Additional recommendations are presented in the section of this report concentrating on CDS 5.0

**Table 1: Proposed Road Map and Key Benefits**

CDS Enhancement	Benefits
Clean tasks and their I/Os in CDS 4.3 and allocate them to the process streams in CDS 5.0. Maintain I/O dependency mapping, eliminate exit criteria, and allocate metrics elements to the task and I/O level of the structure wherever possible.	<ul style="list-style-type: none"> <li>o Ability to analyze critical activity flows across the entire program for specific combinations of metrics.</li> <li>o Pinpointing metrics reporting to explicitly identify the actual work involved at the operating level, and the specific I/O dependencies.</li> </ul>
Introduce "version" and "date" attributes to the I/Os	<ul style="list-style-type: none"> <li>o Support of critical path and what-if analyses based upon risk criteria by simulating the metrics value at the gates and the flow of activities to accomplish given objective</li> <li>o Ability to analyze the trade offs between number of iterations and the risks of missing specific targets</li> </ul>
Extending the decomposition of the I/O dependencies to operational sub-levels while relying on the same structure	<ul style="list-style-type: none"> <li>o Ability to automatically aggregate metrics from the lowest work level to the vehicle level of program tracking using a consistent reporting structure</li> <li>o Ability to pinpoint the risks of development at the operating level for engineering</li> </ul>

# TABLE OF CONTENTS

**CDS: RECONCILING MANAGEMENT OBJECTIVES AND OPERATIONS ..... 1**

**AN INTEGRATED FRAMEWORK FOR MANAGING PRODUCT DEVELOPMENT ..... 1**

EXECUTIVE SUMMARY ..... 1

*Highlights* ..... 1

        CDS 4.3 ..... 1

        CDS 5.0 Targets ..... 2

        Table 1: Proposed Road Map and Key Benefits ..... 3

**CDS: RECONCILING DELIVERABLES WITH OPERATIONAL TASKS ..... 5**

SUMMARY AND RECOMMENDATIONS ON CDS 4.3 ..... 5

*Linking the Phase-Gate Model to Tasks and Work Breakdowns* ..... 6

*Cross-Industry Validation: Alignment with VCOR* ..... 8

    The Maturity of Program Management ..... 9

**CDS 5.0: ALIGNING THE LEADERS IN ENGINEERING WITH MANAGEMENT’S OBJECTIVES ..... 13**

SUMMARY AND RECOMMENDATIONS ON CDS 5.0 ..... 13

*The Implications and Tradeoffs of CDS 5.0 Metrics* ..... 15

*Improving the Accuracy and Understanding of the Risks of Development* ..... 15

*The Risks of Metrics Reporting* ..... 16

*Additional Recommendations* ..... 17

**APPENDIX: VALUE CHAIN MATURITY LEVELS 0-5 ..... 19**

This document is copyrighted © by Collaborative Product Development Associates, LLC (CPDA) and is protected by U.S. and international copyright laws and conventions. This document may not be copied, reproduced, stored in a retrieval system, transmitted in any form, posted on a public or private website or bulletin board, or sublicensed to a third party without the written consent of CPDA. No copyright may be obscured or removed from the paper. Collaborative Product Development Associates, LLC and CPDA are trademarks of Collaborative Product Development Associates, LLC All trademarks and registered marks of products and companies referred to in this paper are protected.

This document was developed on the basis of information and sources believed to be reliable. This document is to be used “as is.” CPDA makes no guarantees or representations regarding, and shall have no liability for the accuracy of, data, subject matter, quality, or timeliness of the content.



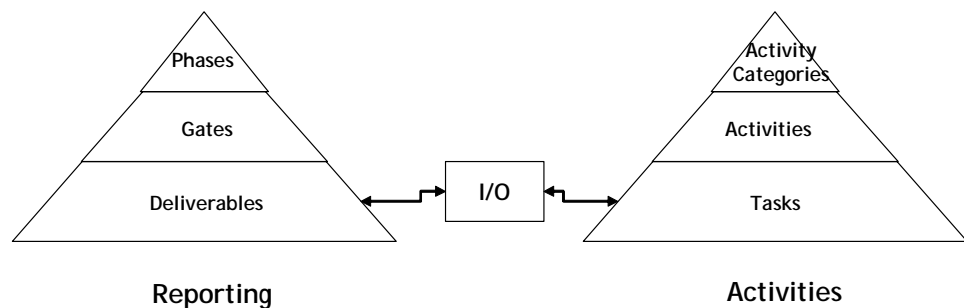
# CDS: Reconciling Deliverables with Operational Tasks

## SUMMARY AND RECOMMENDATIONS ON CDS 4.3

CDS 4.3 is a framework for deliverable-based program management. Although the framework contains modest inconsistencies, CDS 4.3 directly reconciles deliverables and their gate exit criteria with the tasks required to produce those deliverables across all activities and phases of development.

That reconciliation represents a direct and explicit link for management with operations. That is, CDS 4.3 effectively reconciles program reporting for management with tracking activities in operations at an appropriate level of abstraction to meaningfully coordinate and control the product development process. That reconciliation links the management metrics for completing tasks on time with the operational dependencies involved in how those tasks are accomplished. When something is not completed on time, the information is immediately available to determine what must be done to get back on schedule. Taken alone, the stage gate approach simply organizes the deliverables in terms of the targeted completion time. The reconciliation with activities validates the practical realism of the schedule and identifies and sequences dependencies, inputs and outputs. The activity model, PAP (Product Assurance Planning), for CDS contains a comprehensive process knowledge base consisting of over two hundred development tasks linking the deliverables to over nine hundred defined dependencies between tasks.

FIGURE 1  
Structure of the CDS  
4.3 Framework



*A major win, representing dramatic potential for a major payoff, is possible by extending CDS 4.3 to the level of operational work breakdowns that represent the tasks actually taking place in engineering. The extension potentially presents three payoffs:*

- *By reporting down to the operational task level, operational personnel validate and confirm all management estimates for the work being done.*
- *The extension establishes a shared process and common language. Across all disciplines, top to bottom, everyone marches to the same schedule.*

- *A real competitive strength and differentiated advantage, DaimlerChrysler relates the system decomposition of the vehicle to the organizational structure by utilizing cross-functional teams. An extension of CDS 4.3 to the operational task level promotes that approach. The work breakdown then recognizes the dependencies involved with systems interfaces, which are explicitly defined across the entire system, supporting cross-functional collaboration and coordination.*

As targeted in the project, VCOR and the Value Scape tool identified modest inconsistencies that resulted from the manual maintenance of a large knowledge base. Considering the manual maintenance and the size of the existing system, it was surprising not to find even more problems in the knowledge data base.

Even without any extension, automation with supporting software tools represents a mandatory step. That automation should extend to the support of the tools employed at the operational tasks level, such as the creation of Excel worksheets and planning templates for tracking operations. A disciplined review of the appropriate tools available in the market is recommended.

## **LINKING THE PHASE-GATE MODEL TO TASKS AND WORK BREAKDOWNS**

In CDS 4.3, the reporting of progress for a development program is based on the phase-gate model. The deliverables assigned to the gates define the set of information required to pass the gate, to be collected and reported as the outputs of the completed tasks.

The deliverables associated with the gates closely match the tasks assigned to organizational units, which in turn follow the breakdown of the vehicle system using common program reporting semantics across all sub-systems and all development efforts. Thus, deliverable and task designations often get used interchangeably. This creates a minor problem since users may confuse the work that is reported, or what *has been* done, with the tasks assigned to organizational units, or how the work *needs to be* done. Currently, CDS does not provide a definitive set of guidelines on how the work needs to be done, although the PAP4CDS manual references a comprehensive list of best practices associated with different activity categories. The final task breakdown is left to the development teams and sub-system program managers to agree upon.

Reporting on program progress has a minor structural inconsistency, in that milestones are used inconsistently throughout the phases in the programs. The value of these milestones was unclear since they do not help program managers assess the critical path. They are not used for assessing project requirements or for planning resources.

As a targeted objective of the project, however, the application of the VCOR<sup>1</sup> framework identified more inconsistencies with activities. Of 290 tasks reviewed, 251 have targets in terms of explicitly defined information feeding into specific tasks. On average, activities have 3.75 targets, which reflects moderate granularity in the definition of the tasks. However, 13 tasks have over 10 targets. These tasks represent large hubs of I/O dependencies. The largest “task hub,” PV 60 – Develop Vision Paper, targets 22 tasks downstream. That makes it very difficult to analyze a critical path since these hubs create large branches of multiple dependencies. Furthermore, there are 19 tasks that have no targets at all, effectively dangling. Finally, there are 21 circular dependencies, or tasks receiving inputs from the tasks they directly or indirectly feed information into.<sup>2</sup>

The methodology of applying the VCOR framework forced the activities model into a hierarchical structure to explicitly diagram all dependencies. Supported by a relational database, the approach provided a systematic way to analyze the complexity of CDS 4.3 which was represented only in qualitative, lengthy documents without being supported by an explicit hierarchical structure.

Given the inconsistencies identified in the activities model, the implication follows that the evolution of CDS favored management reporting priorities. Most likely, the integrity and consistency of the reporting structure took precedence whenever a compromise needed to be made in the handling of operational activities and tasks.

However, CDS 4.3 shows a resilient structure containing many useful guidelines and a knowledge base for tracking and reporting progress in programs, as well as for supporting the preliminary planning and allocation of resources early in development. Indeed, the '05WK body development program illustrates a consistent application of the dependencies breakdown from CDS 4.3 in detail in a specific domain. In fact, the same structure for breaking down dependencies is followed within the program itself for a multi-level decomposition, starting with the CDS gates and deliverables and extending to the assignment of sub-deliverables at several levels, down to lowest level of granularity. Thus, the consistent application of the approach that provides management visibility can be extended all the way down to the operational levels in terms of tasks such as engineering drawing release, or the sign-offs for metal forming tool tolerances.

Overall, CDS 4.3 represents a common language across all disciplines offering a well abstracted framework for program tracking to higher levels of management. Particularly impressive, its structure may be extended to replicate a full decomposition, to cover requirements for operational tracking while simultaneously aggregating that information with a common reporting mechanism. CDS 4.3, however, does not explore the automation aspects of that alignment, enabling aggregation and disaggregation of tasks and their I/Os. Program

---

<sup>1</sup> Value Chain Operational Reference model

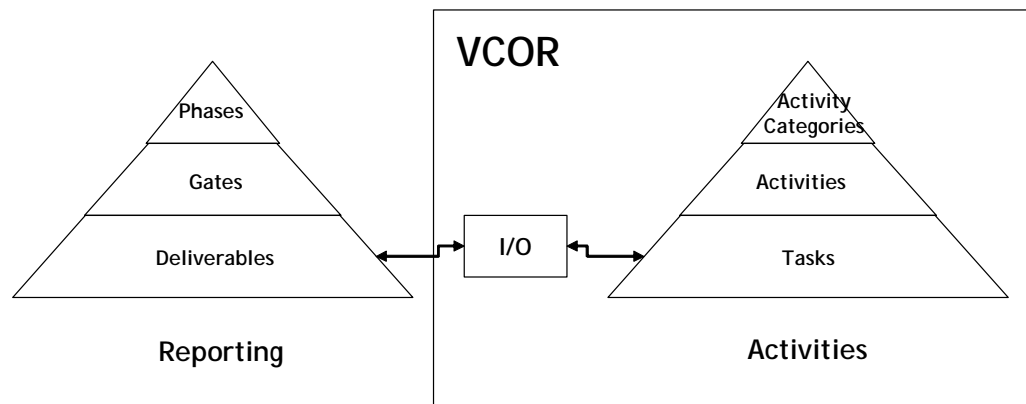
<sup>2</sup> These results cover 80% of the data because the analysis was done on an incomplete version of the data. Roughly 20% of the task inputs were not included in the data set.

managers rely on the knowledge base for preliminary planning, and for defining the phasing of deliverables. Ironically, they then turn to spreadsheets to manually track detailed tasks at the operational level. At minimum, CDS 4.3 should be extended to feed Excel worksheets and planning templates.

Furthermore, different functional groups agree on common deliverables at a relatively high level of abstraction. But the reconciliation of the different views does not extend to the operational level. As a result, each discipline continues to rely on its own flavor in terms of terminology and practices. CDS 4.3 may be extended to normalize the semantics, getting everyone on the same page with improved coordination and cross-functional collaboration. To accomplish that objective, CDS 4.3 would have to be significantly extended, to classify and categorize details of different activities and various types of I/Os at a deeper level of the work granularity. This could be further enhanced with a comprehensive classification of the I/Os and domain-specific configurations following the system breakdown, e.g., body, chassis, powertrain, electrical, and interior.

## CROSS-INDUSTRY VALIDATION: ALIGNMENT WITH VCOR

FIGURE 2  
Cross-Industry  
Validation: Alignment  
of CDS with VCOR



The VCOR framework was developed outside of the automotive sector, with a major contribution from leading electronic firms such as Intel. Of particular note, structurally, CDS aligns very well with the VCOR framework, which validates both CDS 4.3 and VCOR as “reference models.”

Specifically, VCOR recognizes a similar structure for the decomposition of the development process down to a low level of granularity, supported by the explicit definition of I/O dependencies. CDS, however, does not cover the higher abstractions of level 0 and level 1 of the process definition specified in VCOR’s enterprise framework. (These levels cover business processes across the value chain, and macro processes of an Enterprise, as summarized in the Appendix.) That is, CDS focuses specifically on product development, and represents a specific automotive OEM configuration of the VCOR level 3 elements. Indeed, CDS Activity Categories correspond to the VCOR model level 3 elements that follow from the decomposition of VCOR level 1 elements. Level 3 covers “process elements” in VCOR, and “Activities Categories” in CDS 4.3.

Furthermore, the metrics structure in CDS 5.0 corresponds well to the VCOR metrics structure, although CDS 5.0 defines only one level of the activity model as Process Streams, which are equivalent to the VCOR level 3 elements.

VCOR does provide metrics for activities. Although CDS 4.3 does not have metrics, *the VCOR capability in linking metrics and activities confirms that CDS 5.0 should reuse the CDS 4.3 activity model. VCOR already demonstrates how metrics may be associated with the activity model.*

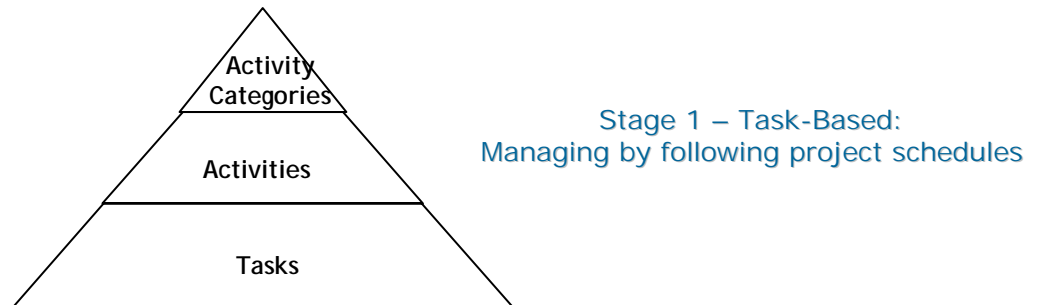
On the other hand, VCOR does not support the reporting structure maintained in CDS 4.3 with the phase-gate approach. Yet, phase-gate reporting represents a common approach in the development environments across industries.

Neither CDS nor VCOR provide a structure supporting decisions that focus on the critical tradeoffs during the product development process, their sequence dependencies, any correlations between decision criteria, and alternatives to metrics or I/Os produced by the activities. Yet the optimization of the sequence of decisions given the context-specific objectives encountered at the operational level delivers significant added value. The metrics assessing program values need to be defined as operational targets to optimize the sequence of tasks. The decision matrices and techniques for modeling were illustrated (Pugh, MCDM, etc.) which may be applied at the operational level. The approaches represent another major potential extension for CDS that could not be fully explored in this project given time and budgeting constraints.

### THE MATURITY OF PROGRAM MANAGEMENT

Program management typically progresses through several phases of maturity that may be grouped into four stages to illustrate the differing requirements for a process framework.

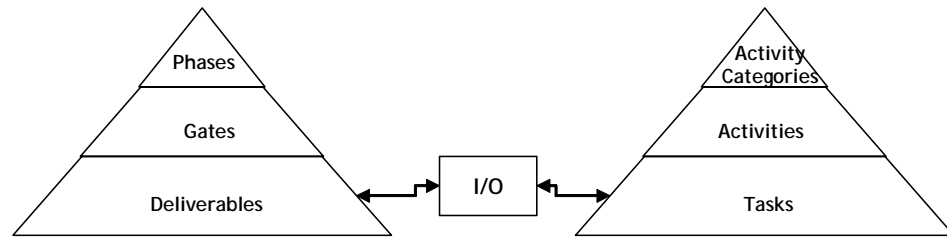
**FIGURE 3**  
*Structure of the Framework  
Supporting Stage 1 of  
Program Management  
Maturity Model*



Stage 1 covers a relatively small scope for the development efforts where the “project equals program.” Iterative development loops are not addressed. Commonly applied approaches such as CP/M (Critical Path Method) and PERT (Program Evaluation Review Techniques) generally apply well only to predictable and repeatable processes. A time view of the program is directly embedded in the activity view of the program. As long as the activity view displays a high-fidelity

coherent picture at the proper level of abstraction for decision making, there is no need to decouple the two.

*FIGURE 4*  
*Structure of the*  
*Framework Supporting*  
*Stage 2 of*  
*Program Management*  
*Maturity Model*



**Stage 2 – Deliverable-Based:**  
**Managing by controlling availability of deliverables at gates**

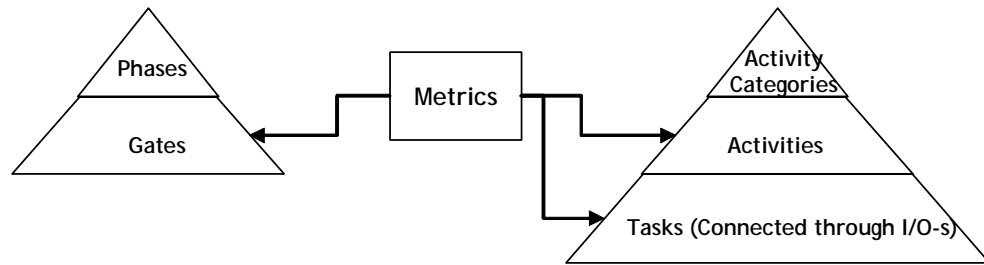
The introduction of stage gates decouples time elements from activities. Each may be managed separately, and may be flexibly aggregated to support iterative development patterns. Development programs typically possess much greater uncertainty involving complex work breakdowns across multiple areas that can be represented with high fidelity using Gantt charts. The approaches represent a step up to stage 2 in terms of maturity. Deliverables at a higher level of abstraction are consolidated and grouped for reporting at the predetermined time intervals. Intricacies of the iterative work loops and dependencies are hidden from the reporting structure, yet are traceable in the complementary activity structure, where I/Os between activities maintain real dependencies to represent the sequences, as well as the iterative loops.

The CDS 4.3 framework enables maturity stage 2.

Stage 3 of maturity introduces metrics reflecting the value of different activities to the business. Metrics effectively translate deliverables into a value scale. The approach is required in companies that assume significant financial risks directly impacted by the results of product development. The amount of work being done, as represented by deliverables, does not equal the amount of progress required to contain the risks of a development program. More realistic metrics that directly reflect financial risks, rather than tasks or time allocated are needed. Even so, this stage of maturity also directly models the dependencies among activities to maintain the control of the activities. Indeed, stage 3 requires that model as a supporting tool.

CDS 5.0 aims at maturity stage 3. Ironically, 5.0 as a work-in-progress does still need to explicitly define development dependencies. As noted earlier, 5.0 should reuse the CDS 4.3 activity model that has already been developed and validated. Moreover, VCOR already demonstrates how metrics may be associated with the activity model, which confirms the merits of reusing the 4.3 activity model.

**FIGURE 5**  
Structure of the  
Framework Supporting  
Stage 3 of  
Program Management  
Maturity Model

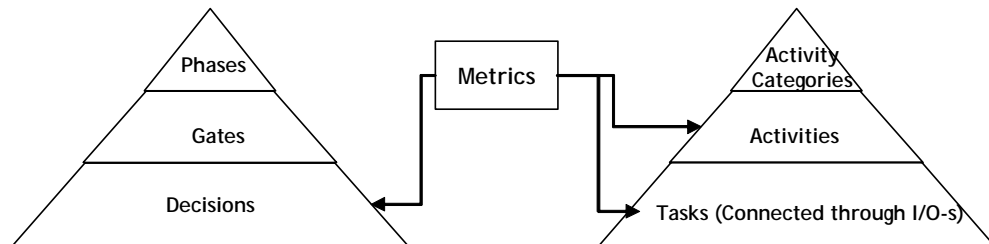


**Stage 3 – Metrics-Based:**  
Managing by measuring effectiveness of the effort at each gate

VCOR currently enables stage 1 of maturity of program management, with extensions to cover metrics. By incorporating a phase-gate capability, VCOR would enable maturity stage 3.

The greatest payoff derives from the ability to assess different scenarios top to bottom, and across all functions. Stage 4 maturity serves those companies with a portfolio of many development options that want to fully maximize the contribution from each as fast as possible. Thus, trading off between different alternatives needs to be a proactive exercise based on precise what-if scenarios of development risks and other product portfolio-related criteria. These analyses help determine the sequence of decisions that will accomplish the fastest progress in arriving at the critical knowledge required to determine the value of each of the programs and their associated risks.

**FIGURE 6**  
Structure of the  
Framework Supporting  
Stage 4 of  
Program Management  
Maturity Model



**Stage 4 – Decision-Based:**  
Managing by objectives from gate to gate

Each stage of maturity requires an accompanying framework structure to accommodate the correlations between the critical elements of reporting and activity management. Without adding decision elements to the framework, neither CDS nor VCOR can support maturity stage 4 of program management. The extensions would support the reconciliation of the metrics for scenario planning with the activities for the critical path assessment covering risks and value added metrics.





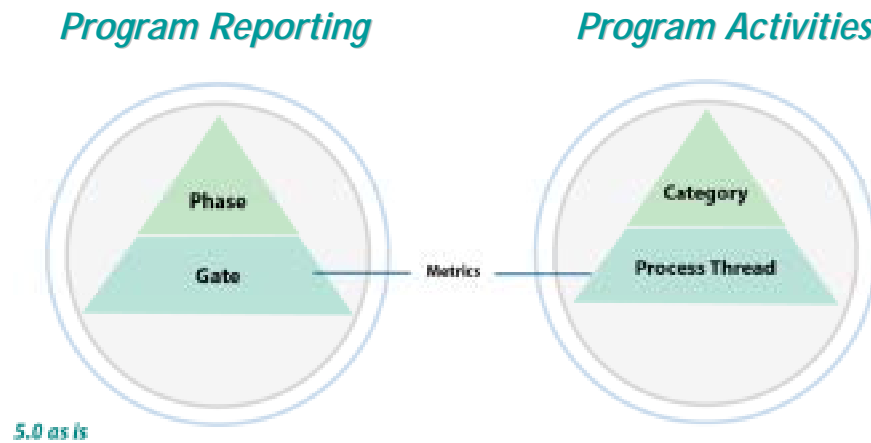
# CDS 5.0: Aligning the Leaders in Engineering with Management's Objectives

## SUMMARY AND RECOMMENDATIONS ON CDS 5.0

CDS 5.0 defines a framework for metrics-based program management. For that purpose, it incorporates a reporting template based on gates for a comprehensive set of metrics to evaluate both effectiveness and efficiency of the development effort. CDS 5.0 explicitly defines streams of activities directly contributing to the specific sets of metrics as they progress through the development phases.

With the transition from a deliverable-based program management to a metrics-based program management, DaimlerChrysler is trying to strengthen accountability and to sharpen the focus of key decision makers on the most critical performance gauges. The transition represents a natural step in the maturation of program management.

FIGURE 7  
CDS 5.0

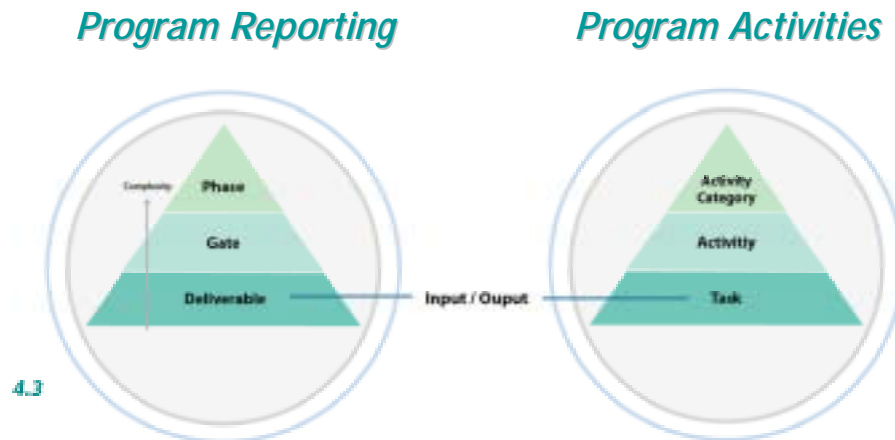


CDS 5.0 also needs to provide a set of guidelines for explicitly identifying and managing critical dependencies during the development process, which was the strongest contribution of CDS 4.3. CDS thereby maintains the link between operational engineering and management. By maintaining deliverables along with the associated reconciliation with tasks and I/Os, CDS 5.0 would support an accurate and complete understanding of program risks. Moreover, that effort would also facilitate the transition from deliverable-based to metrics-based reporting. The “task definitions,” which become process streams in 5.0, are currently at too high a level of abstraction to define dependencies. They resemble the “task hubs” found in the CDS 4.3 structure that require corrective action.

A critical strength of CDS 4.3 is that it aligns I/O dependencies to tasks almost on a one-to-one basis. That reconciliation maintains the balance between higher-

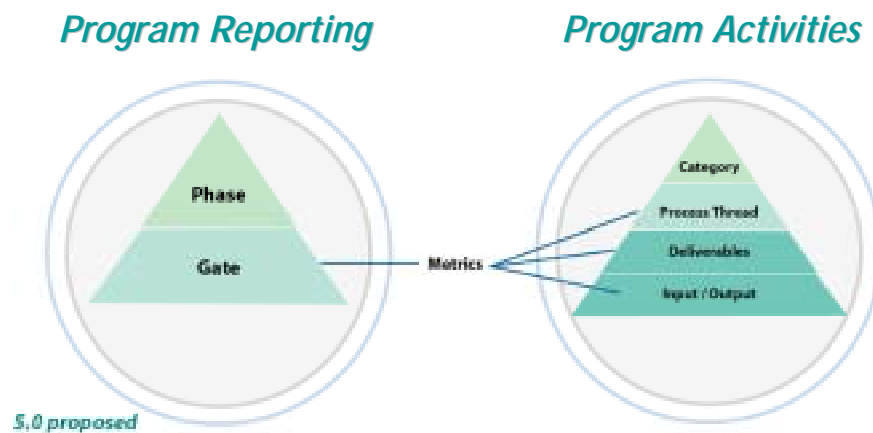
level management reporting and operational needs in organizing activities. Indeed, the reconciliation directly links management with operations. With the dependencies among tasks defined through I/O patterns, the I/O was grouped into deliverables for reporting purposes. Maintaining deliverables in 5.0 supports the reconciliation with the activities structure – and with operations.

FIGURE 8  
 CDS 4.3



It is highly recommended that CDS 5.0 incorporate the alignment of I/O dependencies with tasks already available in 4.3. That means adding the tasks and deliverables to the process streams. The explicit definition of I/O dependencies supports reconciliation and validation, which is even more critical. The dual structure covering metrics and operational tasks still supports the metrics reporting approach, with the comprehensive gate templates already developed. Indeed, the reconciliation of dependencies for activities between different groups strengthens the understanding and operational payoff from applying the metrics.

FIGURE 9  
 CDS 5.0  
 Proposed



## THE IMPLICATIONS AND TRADEOFFS OF CDS 5.0 METRICS

Metrics in CDS 5.0 cover three types: gate entry, gate reporting, and program metrics. This distinction reflects the importance of the different metrics in not meeting the defined targets. It promotes a focus on the gate entry metrics and their critical values as the teams prepare for the subsequent gate reporting at the scheduled time. The gate entry metrics instill a sense of urgency to address critical activities directly.

Relations and dependencies between metrics must be fully defined in CDS 5.0, which extends the current grouping of metrics in streams of process activities that relate to their values. Otherwise, program managers may all too often discover that their specific targets and their responsibilities depend upon independent efforts outside of their control and influence. Then, the targets might not be accomplished without fulfilling unspecified dependencies on inputs from other outside streams of activities. Thus, if the dependencies among tasks were no longer defined through I/O patterns, then CDS 5.0 would not support explicit guidelines to rationally assign responsibilities. Also, CDS 5.0 has not tested and validated in terms of maintaining the integrity of the alignment between metrics and process streams. The metrics will likely appear arbitrary and subjective operationally without that validation.

Even if the grouping of the higher level process streams completely eliminates the dependencies across areas by containing all true dependencies within the process streams themselves, these streams would be too high a level of abstraction to serve operational needs. The internal dependencies between metrics within the process streams would still need to be defined operationally in order to understand and validate the metrics. Of particular value, the direct link between higher level management and operations promoted in 4.3 needs to be maintained.

At best, the extension may cover the gate entry metrics, since the domain of coverage of all other metrics is very wide. Thus, the current structure would then serve the needs for gate-entry metrics, as well as reconcile and validate gate reporting and program metrics. Given the nature of product development, that deficiency then impacts downstream gate entry metrics as well.

## IMPROVING THE ACCURACY AND UNDERSTANDING OF THE RISKS OF DEVELOPMENT

*Metrics may promote a sense of operational urgency, improve focus, and drive proactive risk management.* People better understand the criticality of their efforts when phase gate targets establish explicit schedules. Their motivation in meeting the targets rises when the metrics align to explicit and achievable goals. That is a major plus for introducing metrics operationally.

*However, metrics that involve externally controlled dependencies, or worse yet, unknown and unspecified dependencies, de-motivate and confuse people.* In these cases, clever

operational circumvention thwarts the whole process. Unfortunately, development processes routinely involve intricate cross-functional and cross-domain dependencies across many iterations of creative work. The process in itself creates significant knowledge in eliminating and/or reducing the risks of meeting targets. Knowledge creation happens in successive loops of synthesis and analysis of various aspects of the product value proposition covering functions, features, costs, performance, quality, or service. To maximize and leverage knowledge, development teams need to identify the information sharing required by explicitly tracking the dependencies for that information, the sources of the inputs and the targets for the outputs, and the optimal sequencing. That tracking represents a critical step in promoting knowledge sharing, and empowers operational personnel with the means to continuously improve on their efforts. Motivation and urgency rise directly with pragmatic and visible potential for improvement.

*These dependencies are embedded in the architecture of the system being developed in terms of both the total vehicle and its subsystems.* Crossing multiple functional domains, they appear in physical and functional interfaces, as well as in the practices, tools, and methods of engineering. They cannot be explicitly identified or understood with a deep contemplation of metrics considered alone. Moreover, a significant knowledge base exists in CDS 4.3 that can be reused and further refined. Otherwise, if each team is left to discover and evaluate dependencies on their own without a comprehensive approach targeting multiple levels, they will likely follow their own subjective path which fits their understanding of the metrics criteria. They would then form a relatively narrow point of view related to addressing their specific responsibilities. By reconciling dependencies with deliverables, CDS 5.0 can avoid this double dose of subjectivity which would involve both interpretation of the metrics, and understanding the broader dependencies outside of their immediate area of expertise required to achieve the targets.

## THE RISKS OF METRICS REPORTING

*Metrics drive behavior.* High level metrics that do not explicitly recognize task and I/O dependencies tend to encourage a culture that elevates individual “heroism.” Given clear high-level objectives, when left to their own devices those heroes tap unparalleled creativity and resourcefulness. Paradoxically, that culture characterized a major differentiator of American manufacturing during the industrial age. For large companies as distinct from entrepreneurial startups, a process management framework built on promoting such a culture succeeds only when the culture can be scaled. Two difficulties arise:

- Knowledge contained and gained during the process remains tacitly embedded in the heads of a selected number of individuals, the heroes who may even hoard rather than share the knowledge. Moreover, the expertise can only be applied within the same context to the same or similar problem, involving the same people, with the same approach, because of the lack of documentation. The knowledge would be fully reusable only within the same team working on similar problems under similar circumstances. That may be fine with startups when a program is

completely new and does not benefit from reuse or mass customization approaches.

- Realistically, development efforts today are now dispersed globally, and are spread over many parallel programs. There simply are not enough heroes to cover all programs. Far more systematic efforts are required to reinforce collaboration and support knowledge reuse that bridges and links management and operations to meet the massive scale of the automotive sector today.

*An enhanced CDS 5.0 promotes the culture of collaborative team efforts.* Otherwise, a framework based solely on metrics tends to encourage a culture of complete team autonomy in unconstrained and uncoordinated pursuit of solutions to creative problems. The approach may even backfire, because implicit dependencies and their inertia will not be easy to address when they are ignored in the framework itself. Program management and engineering may collide in assessing priorities and feasibilities.

*Full implementation of a metrics-based framework incorporating the 4.3 capabilities for explicitly identifying and managing critical dependencies extends the existing effort smoothly.* It builds on the existing culture and behavior while migrating from a deliverable-based program management to the new environment. The integration of 4.3 capabilities reconciles the reporting for management and operational oversight, rather than relying on dual reporting for the record and for running the program. The integration also builds on a communal acceptance of a global mission. The leaders of the engineering community will recognize the alignment and validation of their actual work dependencies directly with management's objectives.

## ADDITIONAL RECOMMENDATIONS

The major recommendation in this review of CDS 5.0 focuses on the alignment of I/O dependencies with tasks in CDS 5.0 which are already available in 4.3. The explicit definition of I/O dependencies supports reconciliation and validation. Four additional steps should be considered:

- CDS 5.0 categorizes metrics according to the reporting behavior at each gate. The metrics should also be reconciled with the tasks, and with their I/O's dependencies. Moreover, the template of the metrics should explicitly list the variables of those I/Os that are used for metrics. It should be possible to map over half of the metrics to tasks and their I/O dependencies. Other metrics may involve calculations based on a complex knowledge-based algorithm. At minimum, CDS 5.0 should reference the best available methods for determining these metrics, although it may not identify all variables required across the spectrum of supported I/Os. Any other metric that does not belong to these two types should not be required in CDS 5.0. It will add to the clutter of reporting without clear understanding of what may be in the critical path of accomplishing it.

- Deliverables and the associated information inputs and outputs should specify the targeted date of availability in terms of required, planned, and actual needs. Progress should be tracked through all iterations in the development process.
- Software tools are required to reasonably support the complete CDS 5.0 framework given its growing complexity for the administration of the framework. Moreover, the tools should promote the collection of program data for “what-if” scenario planning and process analysis. A well defined tool reduces the overhead of program reporting, and improves accuracy for risk assessment and decision making.
- A road map for the evolution of the CDS 5.0 might be defined that incorporates three phases, with each targeting new thresholds in maturity:
  - The first release could reconcile the metrics at each gate with the deliverables of activities and their associated inputs and outputs to explicitly identify dependencies. The approach maintains the critical contribution of CDS 4.3 in linking and reconciling management directly with operations.
  - A second release could introduce date reporting and version attributes to the deliverables information. The data serves the analysis of critical paths and what-if scenario planning for major program changes such as the evaluation of cost reduction alternatives or changes in product features.
  - A third release could cover the decomposition of tasks with their associated deliverables and I/O dependencies for specific domains at a detailed operational level for sub-systems development efforts. Decomposition could cover as much depth as justified by simply extending the same approaches and structures already developed. Reporting then may automatically aggregate at any level of activity for metrics analysis and what-if planning.



## Appendix:

# Value Chain Maturity Levels 0-5

### **Level 0 – Value Chain** (*does not exist in CDS 4.3*)

A collection of business processes, which are inter-dependent and create value for the extended enterprise and competitive positioning.

### **Level 1 – Macro Processes** (*does not exist in CDS 4.3*)

The macro business processes of an enterprise. The functional definition of each element is general in nature and represents the activities including research, development, marketing, and operations of a product or service.

### **Level 2 – Possible Process Configurations** (*CDS 4.3 is one of these configurations*)

The level where specific process elements are assigned to an applicable business strategy.

### **Level 3 – Process Elements** (*in CDS 4.3 structure, called Activities Categories*)

The lowest generally applied level of the framework. Although general, Level 3 processes are detailed enough to link to Level 4 activities.

### **Level 4 – Activities** (*same in CDS 4.3 structure*)

A decomposition of a Level 3 process, each Level 4 activity is specific to an enterprise that may or may not be shared among partners.

### **Level 5 – Actions** (*in CDS 4.3 structure, called Tasks*)

Individual work instruction items that produce work deliverables.

(See illustration on following page.)

