Concurrent Process Mapping, Organizations, Project and Knowledge Management in Large-Scale Product Development Projects Using the Design Structure Matrix Method

by

Antoine D. Guivarch


Submitted to the Engineering Systems Division & the Department of Mechanical Engineering in Partial Fulfillment of the Requirements for the Degrees of

Master of Science in Technology and Policy and Master of Science in Mechanical Engineering

at the Massachusetts Institute of Technology

June 2003

© 2003 Antoine D. Guivarch. All rights reserved

The author hereby grants to MIT permission to reproduce and to distribute publicly paper and electronic copies of this thesis document in whole or in part.

Signature of Author

Antoine D. Guivarch
Technology and Policy Program, Engineering Systems Division
Department of Mechanical Engineering
May 16, 2003

Certified by

Daniel E. Whitney
Senior Research Scientist
Center for Technology, Policy, and Industrial Development
Thesis Supervisor

Accepted by

Daniel Hastings
Professor of Aeronautics and Astronautics and Engineering Systems
Director, Technology and Policy Program
Chairman, Committee for Graduate Students

Accepted by

Ain A. Sonin
Professor of Mechanical Engineering
Chairman, Graduate Committee, Department of Mechanical Engineering
Concurrent Process Mapping, Organizations, Project and Knowledge Management in Large-Scale Product Development Projects Using the Design Structure Matrix Method

by

Antoine D. Guivarch

Submitted to the Engineering Systems Division and the Department of Mechanical Engineering on May 16, 2003 in Partial Fulfillment of the Requirements for the Degrees of Master of Science in Technology and Policy & Master of Science in Mechanical Engineering

ABSTRACT

Sustainable success in product design and development relies not only on technical expertise and creativity within the company but as crucially, if not more, on an intelligent design of the development process, an appropriate and dynamic management of organizations, a realistic and disciplined project management, and on efficient knowledge generation, conservation and distribution techniques. These non-engineering skills pose serious challenges to companies designing complex systems like airplanes or automobiles. As these systems have gotten tremendously more complex, their design has kept involving more people, from different working cultures inside and outside the company, all within tighter time constraints. Adaptation to this new context of product development has nevertheless often been very slow because of persistent corporate traditions inherited from the past.

The purpose of this thesis is to demonstrate that Process Mapping and Improvement, Organizations Management, Project Management and Knowledge Management can be reconciled and performed all at once using the Design Structure Matrix (DSM) Method, enabling large and relatively easy improvements of the design activity’s efficiency.

The state-of-the-art in each of the four mentioned fields is first reviewed. The methodology used throughout this thesis, the Design Structure Matrix (DSM) is then presented. The DSM method and some issues of knowledge management are illustrated in a short case study conducted in January 2002 at PSA Peugeot-Citroën in Paris, France. The promising unifying benefits of the DSM method are then thoroughly described through a large project that took place in Summer 2002 at Ford Motor Company in Dearborn, Michigan. It exhibits how DSMs can provide permanent system-level knowledge, guide the design practitioner through a complex process that would hardly be understood otherwise, enable a dynamic management of organizations and open opportunities for process improvement and redesign. The lessons learned finally lead to recommendations on the practice of the DSM method as well as product development in general.

Thesis Advisor: Dr. Daniel E. Whitney
Title: Senior Research Scientist, Center for Technology, Policy and Industrial Development
BIOGRAPHICAL NOTE
Antoine David GUIVARCH
antoine.guivarch@alum.mit.edu

13 May 1979  Born in Bron (69), France.

Education
1996-1998  Lycée du Parc, Lyon, France
            Post-secondary preparatory program for the competitive examinations to enter the
            French Engineering Schools (Grandes Ecoles)

1998-2001  Ecole Polytechnique, Palaiseau, France
            Diplôme d’Ingénieur (Master of Science)
            Majors in Mechanics of Materials & Structures and Economics

2001-2003  Massachusetts Institute of Technology
            Master of Science in Technology & Policy
            Master of Science in Mechanical Engineering
            Research area: Product Development

Professional Experience
            Management and teaching in three-month resocialization internships for offenders

Summer 2000  Franco-Chinese car manufacturer Dongfeng-Citroën, Wuhan, China
             Internship on the final assembly line

Summer 2001  French Embassy, Djakarta, Indonesia
             Analyst: Measurement of financial restructuring in the Indonesian industry

Research Interests
            Management of Product Development
            Automotive Engineering
            Operations Strategy
            Government regulation of industry and politics in general

Awards
2001-2002  Fulbright Fellowship
ACKNOWLEDGEMENTS

This research opportunity and the course of my studies at the Massachusetts Institute of Technology were supported by NSF ERC Grant No EEC-9529140, the MIT Center for Innovation in Product Development (CIPD), the Fulbright Fellowship delivered by the Franco-American Commission for Educational Exchange, and a fellowship awarded jointly by the Fondation de l’Ecole Polytechnique and the Renault Group.

I would like to show great gratitude to Dr. Daniel E. Whitney who supported me throughout this research and taught me much of what follows in this thesis. The flexibility he gave me in conducting my research and his help to find opportunities of application of our research in the automotive industry made this experience both intellectually exciting and professionally formative. The support of the faculty, staff and students of the Center for Innovation in Product Development was also key to create dynamic and convenient working conditions.

Case studies are the backbone of this work. The participation of many engineers to this work, most of them in spite of an extremely busy schedule, enabled extremely fruitful learning.

I thank all participants and supports to the PSA Peugeot-Citroën case study: in alphabetical order, Stéphane BEDDOK, Catherine BOULANGER, Emmanuel COMBES, Pascal DANEAU, Jean-Martin FOLZ, Jean-Louis GREGOIRE, Nicolas GUERIN, Jacques LAEFFEUR, Barthélémy LONGUEVILLE, Anthony PROUST, Janette RIMAU and Thierry VILLARD.

Likewise, I thank warmly the numerous people working at Ford Motor Company who helped me think about process improvement and helped make it happen: in alphabetical order, Nada ALI, Leigh BARRETT, Larry BERARD, Paul BERRY, Robert BLUHM, Dhimant CHHAYA, Tachih CHOU, Thomas ELSNER, John FRANCIS, Dave GRAHAM, Paul HAYNES, Qi HOMMES, Edward HERNANDEZ, Mike HERR, Kristin JOHNSTON, Venu Gopal KOTHAPATI, Michael LAYMAN, Thomas MCCARTHY, Keith MIAZGOWICZ, Chris MINDER, Nceor MOHAMMED, Richard MORTON, Thomas PLOCH, Dustin RATLIFF, Stephen RUSS, Kevin SHEA, Agus SUDJIANTO, Mary VOIGT, Dan WILYARD, Rainer WODE and Tony ZAMBITO.

Finally, I wish to thank so much my family and friends for their continuous support and the happiness they bring in my life. My dearest thoughts go to my mother Annie, my girlfriend Lada and my newborn nephew Abel to whom I wish the happiest life. I wish my father could see his family continue to grow and enlighten us with his thoughts.
# TABLE OF CONTENTS

**ABSTRACT**

3

**BIOGRAPHICAL NOTE**

4

**ACKNOWLEDGEMENTS**

5

**TABLE OF CONTENTS**

6

**LIST OF ILLUSTRATIONS & TABLES**

10

**ACRONYMS**

12

## 1 INTRODUCTION

1.1 Problem Statement

1.1.1 Large-Scale

1.1.2 Product Development Processes

1.1.3 Management Issues

15

1.2 Thesis Goals

17

1.3 Product Development Context

1.3.1 Market power shift from manufacturers to customers

1.3.2 Increasing Product Complexity & Outsourcing

1.3.3 Cost and time-to-market competition

1.3.4 Weight of the past and resistance to change

1.3.5 Unsatisfactory understanding of the specificity of product development

17

1.4 Methodology: the Design Structure Matrix

20

1.5 Research Settings

21

1.6 Preview of the rest of the thesis

22
PART I: CONCEPTUAL WORK

2 LITERATURE REVIEW

2.1 Process Mapping and (Re)Design
   2.1.1 Process Mapping Techniques
         A. PERT charts – Program Evaluation and Review Technique
         B. Integration Definition for Function Modeling (IDEF3)
         C. Gantt charts
         D. The Design Structure Matrix (DSM)
   2.1.2 Incremental Process Improvement vs. Process Reengineering
   2.1.3 Product Development Process Design to Manage Risks

2.2 Organizations Management

2.3 Project Management

2.4 Knowledge Management

2.5 Chapter Summary

PART II: METHODOLOGY & APPLICATIONS

3 METHODOLOGY – THE DESIGN STRUCTURE MATRIX

3.1 The DSM basics

3.2 Different types of DSMs and their applications

3.3 Numerical DSMs and other advanced applications

3.4 The “Whitney Ratio”

3.5 Chapter Summary

4 APPLICATIONS: HYBRID VEHICLES AND ACCESSORY DRIVE SYSTEMS AT PSA PEUGEOT-CITROËN

4.1 Background
   4.1.1 Design process at PSA
   4.1.2 Choice of the System studied
   4.1.3 Analysis objectives
   4.1.4 General description of the conventional thermal FEAD system
   4.1.5 Additional functions and constraints for the hybrid version

4.2 DSM analysis
   4.2.1 Analysis framework
4.2.2 Results and conclusions

4.3 Knowledge Management practice at PSA
   4.3.1 Outline of the work of Barthélémy Longueville
   4.3.2 Observations on Knowledge Management at PSA

4.4 Chapter Summary

PART III: PRACTICAL WORK

5 DESIGN PROCESS ANALYSIS AND IMPROVEMENT AT FORD MOTOR COMPANY

5.1 Background
   5.1.1 Ford Product Development System (FPDS)
   5.1.2 Commodity / product studied: scale
   5.1.3 The need for a process analysis and improvement project:
            Commodity program and task force achievements history

5.2 "Gives and Gets" data collection framework

5.3 "Gives and Gets" analysis results for the commodity design process

5.4 Initial offsite accomplishments & data processing framework to follow

5.5 Deliverable DSM analysis: tracking information flow over time
   5.5.1 Rationale
   5.5.2 Data collection process
   5.5.3 Results and interpretations
   5.5.4 Conclusions so far

5.6 Organizations DSM: enabling a dynamic management of organizations
   5.6.1 Observations on functional organizations’ cultural disconnects
   5.6.2 Organizations DSMs
   5.6.3 Dynamic Management of Organizations
   5.6.4 Who criticizes whom

5.7 Process Improvement Project Next Steps

5.8 Chapter summary and conclusions

PART IV: CONCLUSIONS AND FUTURE RESEARCH

6 CONCLUSIONS AND RECOMMENDATIONS

6.1 Conclusions
6.2 Recommendations on the conduction of process / DSM projects 138
  6.2.1 DSM Building Process 139
  6.2.2 Who leads the DSM project 140
  6.2.3 The value of DSMs 142

6.3 Recommendations on the practice of Product Development 143

6.4 Future Research 144

7 BIBLIOGRAPHY 149

7.1 References 149

7.2 Sponsors’ and support websites 152

APPENDICES 155

Appendix A: PSA Peugeot-Citroën Organization Chart 157
Appendix B: Give and Get data table 158
Appendix C: Deliverables data table 159
Appendix D: Deliverable DSM survey user instructions 160
Appendix E: Excerpts of the colored deliverable DSM 167
LIST OF ILLUSTRATIONS & TABLES

Figure 2.1.1.1: An example of PERT diagram p. 28
Figure 2.1.1.2: Identification of the Critical Path p. 28
Figure 2.1.1.3: The Critical Chain p. 29
Figure 2.1.1.4: Example of an IDEF3 Process Flow p. 31
Figure 2.1.1.5: IDEF3 Object State Transition Network description of the purchase request form in the Material Ordering process p. 32
Figure 2.1.1.6: Example of a Gantt chart p. 33
Figure 2.1.1.7: Some drawbacks of Gantt charts: iterations and overlapping links p. 34
Figure 2.1.2.1: Example of a process reengineering at a car insurance company p. 35
Table 2.1.2.2: Major differences between incremental improvement and process reengineering p. 36

Figure 3.1.1: How DSMs are constructed and read, a simple example p. 44
Figure 3.1.2: Feed forwards and feedbacks in a DSM p. 44
Figure 3.1.3: Partitioned DSM p. 46
Figure 3.1.4: One possible reorganized DSM p. 46
Figure 3.1.5: Task interaction patterns in a reorganized DSM p. 46
Table 3.2: Summary of the four popular categories of DSMs p. 48
Figure 3.4: Consistent average of 6.2 number of marks per row in numerous DSM studies p. 51

Figure 4.1.1: General organization of the design of a car at PSA Peugeot Citroën p. 54
Figure 4.1.4.1: Picture of a FEAD p. 56
Figure 4.1.4.2: Scheme of the FEAD p. 56
Figure 4.1.4.3: Framework chosen for the DSM analysis p. 56
Figure 4.1.5.1: Architecture of conventional thermal vehicles p. 57
Figure 4.1.5.2: Architecture of mild hybrid vehicles p. 57
Figure 4.1.5.3: Conventional mode p. 58
Figure 4.1.5.4: Boost mode p. 58
Figure 4.1.5.5: Inertial engine start mode p. 58
Figure 4.1.5.6: Purely electrical accessory drive mode p. 58
Figure 4.2.2.1: Initial DSM for the conventional thermal FEAD System (40x40) p. 62
Figure 4.2.2.2: Manually clustered DSM for the conventional thermal FEAD System p. 63
Figure 4.2.2.3: Initial DSM for the Mild Hybrid FEAD System (43x43) p. 66
Figure 4.2.2.4: Manually clustered DSM for the Mild Hybrid FEAD System p. 67
Figure 4.3.2: General Structure of OTTs p. 72

Figure 5.1.1: The Ford Product Development System p. 80
Figure 5.1.3: Psychological impact of the FPDS milestone system on the work pace p. 85
Figure 5.2.1: Fictitious example of the input (get) and output (give) scheme declared by team A p. 87
Figure 5.2.2: Example of a get card p. 87
Figure 5.2.3: Example of a give card p. 88
Figure 5.2.4: “Gives and gets” analysis terminology p. 88
Table 5.3.1: List of teams represented at the offsite meeting p. 90
Figure 5.3.2: Wall covered by deliverable cards at the offsite meeting, June 4, 2002 p. 91
Table 5.3.3: Data statistics and bias reduction measures p. 93
Figure 5.3.4: Gives and Gets analysis results for the overall process p. 94
Figure 5.3.5: Framework for the analysis of the outputs of each team p. 96
Figure 5.3.6: Most extreme examples of output distribution in the real process p. 96
Table 5.4.1: Examples of deliverables in the standard format p. 99
Table 5.4.2: Process analysis characteristics and outcomes at this point p. 100
Table 5.4.3: Statistics on the contribution of each team in the overall process p. 100
Figure 5.5.1: Identifying inputs, lead time, perceived quality and impact in the Deliverable DSM p. 102
Figure 5.5.2.1: Deliverable DSM electronic survey layout p. 106
Figure 5.5.2.2: Survey "search engine" control window p. 107
Figure 5.5.2.3: Survey "search engine" input selection window p. 108
Figure 5.5.3.1: Color code for timing status of links and deliverables p. 110
Figure 5.5.3.2: The deliverable DSM diagonal and timing issues p. 111
Figure 5.5.3.3: Geographical distribution of marks in the deliverable DSM p. 112
Figure 5.5.3.4: Timing status of links p. 112
Figure 5.5.3.5: Timing status of deliverables p. 113
Figure 5.5.3.6: Intersection between deliverables that have no predecessors and those that do not feed into any other p. 115
Table 5.5.3.7: Data proving that the low quality high impact assessment is decorrelated from the timing status of the link between deliverables p. 116
Figure 5.5.3.8: Cumulative normalized number of links by timing status over time p. 117
Figure 5.5.3.9: Cumulative normalized number of deliverables by timing status over time p. 118
Figure 5.6.2.1: Information exchange "model" to quantify communication between teams p. 123
Figure 5.6.2.2: Overall organization DSM of the total process p. 124
Figure 5.6.2.3: Overall organizations DSM clustered p. 125
Figures 5.6.3.1 to 5.6.3.5: Organizations DSM for each phase of the process p. 126
Figure 5.6.4: Who criticizes whom org DSM p. 127
Figure 5.7.1: Both views provided to subprocess owners after completion of the assembly and analysis of the deliverable DSM from surveys p. 129
Figure 6.2.1.1: DSM building process as presented in Chapter 3 p. 139
Figure 6.2.1.2: DSM building process with initial collective meeting (Chapter 5) p. 139
Figure 6.2.1.3: DSM building process effectively used in the Ford case study in a situation where disjoined responsibilities suppress interactions between interviews p. 140
ACRONYMS

AVT  Torsion Vibration Damper (*Amortisseur de Vibrations de Torsion*)
CAD  Computer Aided Design
CAE  Computer Aided Engineering
CAM  Computer Aided Manufacturing
CEO  Chief Executive Officer
CIPD Center for Innovation in Product Development at MIT
CPM  Critical Path Method
DEA  *Diplôme d’Études Approfondies*, French graduate diploma comparable to a Master of Science, but more predominantly focused on research
DINQ*  Direction of Innovation and Quality at PSA
DOME  Distributed Object-based Modeling Environment, a research initiative of the Center for Innovation in Product Development at MIT
DPTA*  Direction of Technical Platforms and Purchasing at PSA
DRIA*  Direction of Research and Automotive Innovation at PSA, part of DINQ
DSM  Design Structure Matrix
FEAD  Front End Accessory Drive System
FMEA  Failure Modes and Effect Analysis
FPDS  Ford Product Development System
IDEF3  Integration Definition for Function Modeling 3 (Process Mapping Technique)
IFM  Information Flow Modeling, one of the research initiatives at the CIPD at MIT
KBE  Knowledge Based Engineering
IT  Information Technology
MIT  Massachusetts Institute of Technology
NVH  Noise, Vibration & Harshness
OSTN  Object State Transition Network (part of the IDEF3 Process Mapping Technique)
OTT  Task Technical Flow Chart (*Organigramme Technique des Tâches*)
PAT  Program Activity Team
PD  Product Development
PERT  Program Evaluation and Review Technique
PF  Process Flow (part of the IDEF3 Process Mapping Technique)
STA  Supplier Technical Assistance (a functional team at Ford)
UOB  Units Of Behavior (part of the IDEF3 Process Mapping Technique)

* See PSA Peugeot-Citroën organization chart in Appendix A
1 INTRODUCTION

In the rest of the thesis, unless stated otherwise, the author uses indifferently the expressions ‘Design’, ‘Product Development’, ‘PD’, ‘Product Design & Development’ or ‘Design activity’ to refer to the same broad field of design, as opposed to Finance, Marketing or other functions involved in the manufacturing business. Product Design involving substantial contribution of Manufacturing engineers, Purchasing engineers and managers, Testing engineers, etc., all these activities are embedded in the understanding of the above expressions when they are part of the process of bringing a new product to market. As a consequence, the expressions ‘Design practitioner’, ‘Design world’, ‘Product Development world’ or ‘Design community’ should be understood with the same broad definition.

1.1 Problem Statement

This thesis fits in the numerous attempts to make the product development activity more efficient in terms of duration, cost and product quality. It focuses primarily on the management of large-scale product development processes.

1.1.1 Large-scale

It is important to emphasize that this thesis does not deal with fairly simple and short term processes involving teams of, say, less than twenty members, which are widely studied and documented in business books examples and case studies. It deals with highly complex processes, that involve dozens if not hundreds of people and that last several years from end to end. Of course these large-scale processes pose far deeper and more difficult management problems than what business books can present in two or three pages per case.

1.1.2 Product Development Processes

Three phases of product development process are usually distinguished by literature (see Ulrich and Eppinger, 1995, 2000 for a reference book on the different steps of product design and development): the strategic and planning phase, the execution phase and the deployment phase. The strategic and planning phase mostly deals with identifying customer needs, generating, selecting and testing concepts, as well as budgeting and planning. The execution phase translates the specifications drawn from the strategic phase into a real design, ready for production. The deployment phase involves product release, production ramp up, logistics and distribution.
By Product Development Process, the author refers to the end-to-end set of tasks that constitute the execution phase only.

1.1.3 Management Issues
The focus is limited to the following issues that are particularly challenging in the current practice of the Design activity.

Process Mapping and (Re)Design
As Reinertsen says, the way product development is practiced is “messy”. Not only are PD processes based on “evolutionary adaptations embedded in (...) traditions and rituals”, but most of the time they are not clearly documented and known to the people who perform them. Sometimes, processes are nevertheless documented but in a clean and ideal way, most of the time imposed by higher-level management, that does not reflect how work is done in reality. Frustration and dissatisfaction in design teams result and create even more friction in the process. See Chapter 5 section 5.1.3 for a detailed description of this phenomenon Ford Motor Company.
A clear realistic documented process needs to be available to its practitioners to guide them, serve as an education tool for new recruits, and enable opportunities for process redesign, also called reengineering, through its analysis. The issue in this field is: how to map a process in a way that reflects its complexity to a sufficient extent, and how to collect the necessary data?

Organizations Management
The transcription of the PD process, understood as a set of tasks, into organizational structure confronts persistent historical traditions and mental models, usually including cultural isolation and to some extent distrust to others, which do not facilitate fertile cooperation. Structuring a relevant organization to support a process surely is a challenging task. The questions of how to identify who should work with whom at which moment of the process, and how to translate this time dependent communication network into a human structure are critical.

Project Management
A direct consequence of the issues already identified is that managing and tracking the progress of a PD Process is extremely difficult. The managers need an efficient set of tools to assess where their teams are in the process, where they need to go in the next steps and how. A key aspect is the management of information flows, from whom, to whom, when and with which content, to make sure that everybody gets the information needed on time to keep the project moving forward. As the case study will show (see Chapter 5, section 5.5), PD processes are full of timing disconnects, that add up to the cultural disconnect identified earlier. The question is: how can process progress be tracked and large-scale projects efficiently managed?

Knowledge Management
Because today’s business is increasingly knowledge-centered, knowledge management plays a critical role in PD Processes management. Obviously, several points of the three previously discussed issues overlap with knowledge management: documentation of the process, education of the new recruits, identification of the communication networks and where and when the knowledge is generated, etc. Knowledge Management also covers a broader set of issues: how can knowledge be generated, recorded, shared and updated to build an accumulated asset for the future of the company when design practitioners are already striving to complete their design
Introduction

assignments? How to document and share system-level knowledge that describes the intricate internal interaction and interface patterns that complex systems always comprise and which actually make them so difficult to design? It is important to understand that these concerns necessarily include reflections on Information Technology (IT) and the way information systems are designed, but are not limited to them. Even more challenging is the reflection on the philosophical change that is required in corporations to consider knowledge as an asset that creates value when efficiently accumulated, managed and reused.

1.2 Thesis goals

It could seem that this thesis tackles four problems, or actually four sets of problems outlined above, which is not particularly advisable for a Master’s thesis. Well, obviously, the four fields share some significant overlap. But even further, this thesis intends to show that the four general fields can be unified in one whole and treated as such with a single methodology.

The goal of this thesis is to demonstrate how process mapping and (re)design, organizations management, project management and knowledge management, with all of their attached critical questions listed above, can be addressed concurrently with significant results.

1.3 Product Development Context

Before getting into the detail of the methodology used, it helps to review some of the external and internal factors that have driven the evolution of the Product Development activity in the last decades. The industrial world has been undergoing several major changes of context indeed, that have deeply impacted the way Product Development performs.

1.3.1 Market power shift from manufacturers to customers

Following Michael Hammer, the process reengineering ‘guru’, in Charles Fine’s Operations Strategy class at MIT (15.769, Spring 2003), it is fair to observe that the industrial world has shifted in the past century from a context of mass customer demand with few available manufacturers of equipment goods, to consistently saturated markets, at least in the industrialized countries, with several large manufacturers fiercely competing on the entire planet. As a result, manufacturing companies have gradually lost the power of imposing their product design, features, and lifetime, which they chose so as to make production easier, to the favor of the customer who now drives companies to supply the perfectly customized product to fit personal needs. Indeed, the famous (alleged) Henry Ford quotation about the model T, “you can have any color you want, so long as it’s black” sounds totally anachronistic in the era of personalized cellular phone covers and ring tones. At the time, Henry Ford restricted the choice of color of model Ts because only black paint would dry fast enough to keep up with the takt time of his assembly line. The Product Development world now faces greater challenges in meeting always increasing and extremely demanding customer expectations.

---

1 The takt time of an assembly plant is the speed at which parts must be manufactured in order to satisfy demand
1.3.2 Increasing Product Complexity & Outsourcing
A natural consequence of this market power shift, made possible by technological advance, has been continuously increasing product complexity. We have come from the Ford Model T achieving its simple goal to provide personal transportation to the contemporary car with on board integrated cellular phone, GPS positioning, DVD player, variable cam timing, exhaust catalyst, etc.; from the black and white TV set to the color LCD flat screen with remote control, video conference and internet surfing capabilities; from the basic ice box to the hi-tech refrigerator, freezer, ice dispenser with clock; from the first telephone to the fully customizable cellular phone with large color screen, internet capabilities, caller ID and all kinds of voicemail and call transfer services.

As products get more complex, their design involves more varied competencies, more people, from different technological backgrounds and with different working cultures, which requires much more difficult interface and interaction management and coordination. Some examples of this type of challenges, and sometimes failures, are the recalls that some car companies have been making after receiving customer reports that some electrical functions were not performing well. In some cases they found out that the failure was due to unexpected incompatibility and interference between electrical components that were not expected to have a significant interface, like cruise control and satellite radio for example.

On top of this evolution within companies, a general response of firms to increased complexity and cost competition has been to outsource not only some manufacturing activities but also the design ones to “full service suppliers” for numerous parts or subsystems. This is a revolution for manufacturing companies: they have moved from being manufacturers of items to being system integrators of items they have others make for them. Their activity is therefore much more knowledge intensive than before: the transfer of many activities beyond the borders of the company poses even more difficult challenges to the coordination and communication that product design and development requires. A typical example of this difficulty is the Ford Explorer / Bridgestone Firestone case that came out in year 2000. Complex interaction between the tires and the car created a severe failure to the system: tread-separation. Firestone was first blamed for supplying dangerous tires, which they acknowledged, but further research in 2001 concluded that the Ford Explorer design had amplified the defect and made it so critical (E-center for business ethics, 2002).

1.3.3 Cost and time-to-market competition
In parallel to increasing product complexity, competition has driven continuous cost-cutting and time-to-market reduction efforts. In order to remain competitive, manufacturing companies have to offer a product closest to the customer demand of the moment, with the latest technology, for the lowest possible price with virtually no profit margin. Computer hardware manufacturers have for example tremendously cut the cost and price of their equipment over the past 25 years. While car development time has fallen from five to six years at the beginning of the 90s to two to three years today, General Motors and Ford Motor Company have engaged in the recent years in a suicidal price war by offering zero interest rates credit plans to their customers. They are also allegedly losing money on every compact car (e.g. the Ford Focus) they make, a segment even more sensitive to price competition than others. Obviously, this evolution puts another challenging constraint on the product development activity that can apparently be contradictory to the complexity requirement. Basically, designers are asked to create more complex, more reliable, better products for cheaper and in less time. Definitely not an easy job...
1.3.4 Weight of the past and resistance to change: organizational structure and process

The author is convinced that the design world has been badly handicapped in its response to the challenges outlined in the previous section by very persistent corporate traditions and specialization culture that oppose change with great inertia.

The way product development organizations are structured was inherited from Adam Smith’s theory of the division of labor, which was implemented by Fred Taylor and Henry Ford. In most companies, product development is still structured primarily by functional specialty, i.e. the purchasing department, the manufacturing engineering department, the CAE (computer aided engineering) department, the system design department, the component design department, etc. Human resources from each of these departments are allocated to specific projects in which they work with people from other departments, but the engineers still report primarily to their home department on which the evolution of their career depends. While such a division enabled the development of specialized expertise in companies through specific training and knowledge transmission within specialties, it also let very different working cultures develop, which sometimes makes teamwork very difficult. The traditional old-way opposition between manufacturing engineers and design engineers is the best recognized and documented organizational tension to illustrate that point (Cook, 1992). Although everybody has known for decades that there is no reason for a design engineer and a manufacturing engineer to oppose each other because they both contribute to a successful product, need each other’s expertise and should instead work hand in hand to succeed, the usual distrust and tension is generally still there in numerous large companies. On top of the occasional tension that can exist between particular departments, the fact that departments are fairly independent and self-existent makes collaboration difficult even if there is no historical reason for friction. People working on the same project may be located in different buildings, have different working methods and schedules, and a different understanding of what information other members of the team need from them. As shown in detail later in Chapter 5, the amount of cross-organizational misalignment is often extremely surprising. Different departments simply have very little knowledge on how to relate to other departments, when and for what, although they are all involved in common projects together everyday. Not only is coordination challenging, but any change in the organizational structure faces strong opposition by department heads or representatives who would be likely to lose their power if a more cross-functional structure was to be put in place.

Outdated organizational structures are not the only result of historical heritage: product development processes are also often driven by traditions. Donald G. Reinertsen, co-author of bestseller “Developing Products in Half The Time” (Smith & Reinertsen, 1991), writes in “Managing the Design Factory” (Reinertsen, 1997, p. 2): “In most organizations product development is a mix of black art and esoteric ritual. Most development processes have not been consciously designed, but rather have evolved along a bumpy path. Things are tried that work, so we keep doing them. Things are tried that don’t work, so we stop doing them and never try them again. Our processes contain a record of this evolutionary adaptations embedded in our traditions and rituals. (…) Product Development is messy (…). You may want to enjoy the results, but you don’t want to know how they do it.” The last two sentences very well describe the author’s impressions on the field during the case studies that will be presented in this thesis.
1.3.5 Unsatisfactory understanding of the specificity of product development

The author also believes that efforts to reform the product development business have been somewhat confused by a misleading mental model of product development. It seems that understanding the core original nature of the Design activity has not been totally achieved yet. Namely, Design has the particularity of dealing with information flows and being inherently and irrevocably iterative – new technologies or combinations of technologies need to be tested and are very often too complex to be comprehensively and perfectly predicted on the first design or prototype (see Eppinger et al, 1994 for such description of product development tasks). It makes its process radically different from production, a process dominated by a linear and usually waterfall type of flow of materials. Indeed, the specificities of Design, which challenge our natural preference for clean linear successions of tasks with no feedback and no rework in processes, need to be recognized before any attempt to transfer theories and methods from other fields. Donald G. Reinertsen puts it this way (Reinertsen, 1997): “It is very easy to treat product development as a black art rather than a science, because so many of its elements are unpredictable. This unpredictability is inherent in doing things we have never done before. (…) There are two schools of thought as to how we might get to [a] good design. One school holds that we should strive as developers to reduce the error rates. If we keep analyzing the design to minimize the number of errors, we will get a better design on the first try… The other school of thought says: do it, try it, fix it. This school lacks the moral high ground of the other approach, but is well-grounded in the practical observation of what works for successful companies in the real world.” His book very well shows what can Design learn from the lessons of the factory while still respecting important differences between product development and manufacturing.

Of course, not all the iterations in current product development processes are necessary, the distinction between necessary and unnecessary iterations will be made later on in Chapter 3, section 3.1. But even in a perfectly optimized product development process, iterations and rework loops will still play an important role. The author feels that this core characteristic of product development, the iterativeness, is still not fully understood even by design managers and planners. The proof is that so many companies still make timing plans assuming that system design will be followed by detailed design, prototyping and testing without major loops and finally end up delaying their product release by months if not years when confronted with inevitable but unexpected and unplanned problems. This scenario for example happened recently in Europe to Renault with the Avantime, a new crossover concept of high-end coupe and minivan, whose market release was delayed by more than a year, initially planned for mid 2000 and finally accomplished in the fall 2001. The model actually failed to conquer a new market and production stopped as of February 2003.

1.4 Methodology: the Design Structure Matrix

This thesis employs the Design Structure Matrix (DSM) method to study product development processes. DSMs were created by Donald Steward (Steward, 1981, 1995) and have been enhanced since then in many ways (Sosa, Eppinger & Rowles, 2000; Whitney et al, 1999; Rogers, 1997). DSMs are square precedence matrices that map oriented interactions between elements listed in the same order in rows and in columns. The elements can be physical parts or subsystems, human organizations or individual people, design parameters, design specifications, design tasks, process deliverables, just about anything that is relevant to the study, in the context of this thesis, of product development processes. DSMs, by mapping the interactions between
their listed elements, help identify clusters of elements that are tightly interdependent. Different types of DSMs of course lead to different conclusions. In the case of task DSMs, design loops can be identified, tasks can be resequenced to minimize iterations and smooth the process. In the case of organizational DSM, clusters of organizations that need to work closely together are identified and enable an intelligent structuring of the teams to take organizational interdependencies into account.

1.5 Research Settings

This research was conducted within and supported by the MIT Center for Innovation in Product Development (CIPD) and especially its Information Flow Initiative, led by Dr Daniel E. Whitney who also supervised this work. For a description of the funding sources, please refer to the acknowledgements section p.5.

The DSM method was applied to two case studies with different contexts. Both took place in the automotive industry because it is the author’s field of interest, but the focus of this thesis being process and not product, the lessons learned through the case studies as well as through the whole thesis should be general enough to apply to any development process of any product. The first case studied the design process of the Front End Accessory Drive (FEAD) System of engines at PSA Peugeot-Citroën in January 2002, at the technical centers of La Garenne-Colombes (92) and Vélizy (78) in the surroundings of Paris, France. The purpose was to study the impact on the FEAD system structure (the way its subsystems relate to each other) of the evolution from a conventional internal combustion engine to a mild hybrid engine. A design parameter DSM was used for that purpose. However, the case study lasting only one month, it remained limited to a rather small system and was more an introduction to the real world problems of product development than a full large-scale study that this thesis intends to help conduct. It was the perfect introduction, however.

The second case study took place at Ford Motor Company in Dearborn, Michigan, USA, over the full summer 2002 to fully analyze and improve the entire design process of a complex commodity with all the attributions listed in the problem statement section. The project that was the support of this case study was scheduled to have lasted for over 5 years, involves hundreds of people, and incorporates a high degree of complexity requiring an equally complex product development process. The DSM method was this time used in its full extension, from deliverable DSM to organization DSM, involving interviews, offsite meetings and elaborate electronic surveys over the course of the summer and continuously later on up to the moment of writing of this thesis. This second case study constitutes the heart of this thesis and the principal achievement of its goal, it is the demonstration by itself that Process Mapping, Organizations Management, Project Management and Knowledge Management can be reconciled and handled together through the DSM.
1.6 Preview of the rest of the thesis

The rest of the thesis is organized as follows.

The first part of thesis describes the conceptual background. Chapter 2 successively reviews the state-of-the art of process mapping and (re)design, organizations management, project management, especially applied to product development, and knowledge management as presented in the available literature.

The second part of the thesis introduces the methodology used thereafter and shows a first application through a case study. Chapter 3 presents the Design Structure Matrix, its rationale, its various possible applications and the results it yielded up to this thesis. In Chapter 4, the main characteristics and potentialities of DSMs and some aspects of Knowledge Management are illustrated through a fairly simple case study on the FEAD System design at PSA Peugeot-Citroën.

The third part of the thesis focuses on practical work accomplished to support the thesis goal. Chapter 5 thoroughly presents the analysis of the complex design process of a commodity at Ford Motor Company and the improvements achieved in the fields of process, organizations, project and knowledge management.

The fourth part of the thesis examines the general lessons learned from the case studies and the reasoning of this thesis, and opens for future research. Chapter 6 draws conclusions on the rationale of the thesis, exhibits recommendations both for the practice of the DSM method and for the management of product development processes. It finally lays out opportunities for future research.

Chapter 7 contains all the references and useful links the reader might want to examine.
Introduction
Part I: Conceptual Work
2 LITERATURE REVIEW

This chapter intends to give the reader a flavor of each of the four fields this thesis aims at performing concurrently: Process Mapping and (Re)Design, Organizations Management, Project Management, and Knowledge Management. Collections of books are written for each field, and it would be presumptuous to ambition covering each of them comprehensively. The role of this chapter is therefore to pick from each field relevant ideas that apply to the general context of this thesis. The reader is invited to consult references mentioned below for more information. Since the focus of the main case study chapter, Chapter 5, is the analysis of a large complex product development process, more emphasis is given to the first section on Process Mapping and (Re)Design.

2.1 Process Mapping and (Re)Design

This section presents some notions of process thinking and improvement, not necessarily restricted to the Product Development World. Process is defined as an organized group of related tasks that work together to create value. Typical examples in the manufacturing business world are the manufacturing process (from raw materials to finished product), the order fulfillment process (from the placement of an order to the delivery of the product or the service to the customer) or the product development process (from an identified customer need to a finished product design). The first step of any type of process reflection is to know what the process is and how it works: it needs to be mapped. The discussion then moves on to differentiating incremental process improvement from process reengineering. Finally, an interesting proposal of a method to design product development processes, so that they match risks companies face, is reviewed.

2.1.1 Process Mapping Techniques

This first section is largely based on work completed within a team for the purpose of MIT class 15.769, Operations Strategy, taught by Professor Charles Fine and Dr. Michael Hammer in Spring 2003. The author acknowledges the participation of Dexter Borbe, Dan Park, Justin Sanchez, Stephan Schmidt and Erik Smith to the information presented here.
Modeling and mapping process workflows is an inexact science. Many methods exist that leaders and managers can use to get a better handle on their processes. Some of the most popular ones, which also happen to be project management tools, are presented below.

A. PERT charts – Program Evaluation and Review Technique
PERT is a simple method for modeling and mapping process flows used extensively in project management. Quite simply, PERT diagrams model project process “value chains” – those activities that must be accomplished to ensure successful project completion. Figure 2.1.1.1 below provides a hypothetical example of a very simple PERT diagram for the construction of a factory. Numbers in the captions provide estimates of the time (in days) that it will take to complete discrete process steps.

![Figure 2.1.1.1: An example of PERT Diagram](image)

Source: (Goldratt, 1997) p. 66

Although initially intended for use in project management, PERT diagrams have wider applications in modeling, mapping, and understanding all types of business processes, because they capture process inputs and outputs, which help the person using them understand key relationships – especially dependencies – that exist in process flows. One key advantage of PERT charts is that they can be as simple or as complicated as needed.

The Critical Path Method (CPM)
A key idea that comes from mapping a business process is the idea of “Critical Path”, the path of dependent steps in a process with the longest completion time. In the simple example provided in Figure 2.1.1.1, the Critical Path is 150 days, which follows the path required to build the building, through to the final step of equipment installation: Build Building → Make Building Functional → Equipment Installation. Figure 2.1.1.2 below illustrates this concept.

![Figure 2.1.1.2: Identification of the Critical Path](image)
The Critical Path is extremely important, because it is the sequence of tasks that, when not completed on time, will delay the entire project. It is therefore these tasks that the process (or project in this case) manager needs to focus on since they contribute most towards the successful and timely completion of the process. Understanding the Critical Path helps:

- managers place appropriate focus on the parts which are most crucial to process success,
- the people involved in the process understand the value of the work they do in supporting the critical path,
- people understand how not to impede progress along the critical path.

**The critical path can shift:** if certain parts of the process impede the critical path, then those parts of the process that block progress along the critical path become themselves part of the critical path. In the example provided in Figure 2.1.1.1, if the “Contract Vendors” step takes 45 days instead of 15 days, then the Critical Path is now 165 days long and follows the sequence: Contract Vendors → Build Machines → Equipment Installation.

**The Critical Chain Analysis**

One key assumption of the Critical Path method is that the key constraint to the accomplishment of a process is time. Yet, while the goal of many processes may be the minimization of required time, the limiting constraint may often stem from other resources. When a resource other than time is the limiting factor, then PERT diagrams can still be useful tools to visualize dependent sequences of steps. However, it is now most important to understand how the constrained resource affects process accomplishment. The path required by this most constraining resource is termed the “Critical Chain”.

![Diagram of Critical Chain](image)

**Figure 2.1.1.3: The Critical Chain**

Figure 2.1.1.3 illustrates the concept of the Critical Chain. In this model, the duration of tasks is represented by the length of their box. The sequence of steps that takes the most time to complete is therefore the horizontal path in the middle of the diagram. However, in this example, the limited availability of a key resource X creates a constraint on many tasks in the entire process. An example of this limited resource may be a piece of equipment that the company has only one
or two units of, or the services of a specialist who must process or inspect the task. Without limitation on resource X, the subprocesses above and below the middle sequence of tasks could be performed in parallel because they do not depend on each other and only converge at the end of the process. However, the constraint in resource X makes them sequential de facto, because they now share a relationship through the limitation of resource X. The path marked by the dotted line is the Critical Chain of the process and is the one followed to reach completion. Again, the Critical Chain will shift if another resource common to several tasks becomes more constrained than X.

This is a critical realization as it gives managers and leaders a better understanding of the critical factor that affects process accomplishment. In this case, it is the efficient utilization of resource X. Similarly, this also informs employees of how they should focus their energies. Once people realize that the Critical Chain of events that must take place in a process revolves around the efficient utilization of a key resource, they can focus their energies on making it happen. People that are responsible for completing the shorter subprocesses can understand that they need to ensure the completion of all steps leading up to the resource constrained steps before the assignment of resource X to them. Understanding which resources are constrained and how it affects the process allows managers to allocate that precious resource more effectively. They can also track how the organization as a whole is optimizing utilization of the limited resource.

PERT charts are therefore extremely useful to map processes and attract the attention on their bottlenecks and constraints. However, when relationships between tasks are too numerous and complex, the chart gets harder to read. PERT charts also do not allow iterations and loops between sets of tasks, which is key in product development.

B. Integration Definition For Function Modeling 3 (IDEF3)
Another graph-based method for mapping and modeling a process is IDEF3. IDEF3 captures the temporal information, including precedence and causality relationships between various processes and their associated tasks. IDEF3 process mapping comes in two forms, process flow (PF) and object state transition network (OSTN).

Process Flow
A process flow description captures the knowledge of "how things work" in a process: for example, the description of what happens to a part as it passes through a sequence of manufacturing processes. Specifically, an IDEF3 PF description captures a description of a process and the network of relations that exist between tasks within the context of the overall process. An IDEF3 PF description consists of three building blocks: units of behavior (UOB), links, and junctions. UOBs represent types of happenings or tasks in the process, and are shown as boxes. Arrows indicate the precedence relationships or constraints that hold between the tasks being described by the UOB. Junctions represent points in the process where a process splits into multiple paths, or where multiple paths merge, and are represented by X.

The following example from (Mayer et al., 1995) provides an illustration of a "Material Order" process. A IDEF3 PF description would look as shown in Figure 2.1.1.4. The IDEF3 PF allows the capture of tasks at varying levels of abstraction by using a mechanism called "decomposition". Decomposition is a means of showing a more detailed description of a UOB.
The UOB Request Material in Figure 2.1.1.4 has been decomposed into UOBs 7 through 10. The numbers in the lower-left corner of UOB boxes 7 through 10 include a reference to UOB 1 (the first digit) and the decomposition (decomposition 1 of UOB 1). This IDEF3 numbering approach allows explicit traceability between levels of detail in the description. This type of number approach is critical, since a UOB can have any number of different decompositions, all on the same level. This ability of the IDEF3 to incorporate more than one decomposition for the same UOB enables to represent different points of view of the processing relating to the UOB. After all, the process description depicted in Figure 2.1.1.4 shows the Order Material process from the business owner’s point of view. It is possible to conceive of other points of view for this process: for example that of the Account Manager. Each perspective would be presented in a separate decomposition with a unique label and number.

![Diagram](image)

**Figure 2.1.1.4: Example of an IDEF3 Process Flow (Material Order Process)**

**Object State Transition Network**

The second form of IDEF3 process mapping is object state transition network (OSTN). OSTN summarizes the allowable transitions an object may undergo throughout a particular process. The key elements of an OSTN are object states and state transition arcs. Circles represent object states and arrows connecting the circles represent state transition arcs. The object state is defined in terms of the facts and constraints that need to be true for the continued existence of the object in that state. State transition arcs represent the allowable transitions between the selected object states. It is often convenient to highlight the role of a process or task in a state transition, by attaching a UOB to the transition arc between the two object states. Figure 2.1.1.5 shows a IDEF3 OSTN description for the states of a purchase request form in the Order Material process above. In this case, the purchase request form is the object.
Figure 2.1.1.5: IDEF3 Object State Transition Network description of the Purchase Request form in the Material Ordering process

Advantages of the IDEF3 approach to process modeling and mapping show as it:
• Helps visualize process flow and define clear dependencies
• Captures and presents information at various levels of abstraction through applying levels of decomposition
• Incorporates different viewpoints of process through parallel decompositions
• Portrays the object progression through process stages and the role of relevant tasks

Some of the drawbacks to using the IDEF3 approach to modeling and mapping a process are:
• Gets very complex and detailed, especially if incorporating various levels of decompositions
• Requires an understanding of the IDEF3 methodology to understand the full value of information captured, else it may be confused with a simply PERT chart
• Allows neither cycles and loops between UOBs, nor an object to return to a previous state. IDEF3 is therefore of probably little help to map PD processes.

C. GANTT charts
Gantt charts, named after their creator Henry Gantt (one of the pioneers of Scientific Management with Frederick Taylor) emphasize timing issues. Gantt charts are a project planning tool that can be used to represent the timing of tasks required to complete a process. In a Gantt chart, each task takes up one row. Dates run along the top in increments of days, weeks, months or years, depending on the total length of the project. The expected time for each task is represented by a horizontal bar whose left end marks the expected beginning of the task and whose right end marks the expected completion date. As the project progresses, the actual duration of tasks can be shown on the chart to visualize the impact of the time shift on the rest of the project.
Tasks may run sequentially, in parallel or overlap partially with each other. More precisely, one task will point to another if it serves as an input to the other task. The idea of a sequence can therefore be deduced from the links between tasks. Also, tasks that are not linked together can be performed independently, concurrently, in parallel. More elaborated versions of Gantt charts do not necessarily assume that if task A’s output flows to task B, task B cannot start until A is completely done. This situation is rarely the case in the real world, and an easy way to reflect that is to state that A’s output flows to B but B can start as soon as a certain proportion of A is completed, for example 80%. In this case, tasks A and B can therefore overlap while the last 20% of A is completed. Figure 2.1.1.6 exhibits a classical example of Gantt chart.

![Gantt Chart](image)

**Figure 2.1.1.6: Example of a Gantt chart**

Source: [www.smartdraw.com](http://www.smartdraw.com)

Some of the advantages of Gantt charts include:
- Gantt charts are **very useful for planning** (assessing how long a project should take).
- They also provide a **very instinctive visual representation** of a process that helps employees quickly understand where they are and what the next phases are.
- They also permit a **quick identification of the critical path** in the process.
- They allow tasks to overlap partially, which is often not allowed by other mapping tools.

However, Gantt charts also exhibit some drawbacks for process mapping, such as:
- Gantt charts can be difficult to interpret and understand if there are many tasks which interconnect with each other. The links between tasks may either not be shown (which makes it hard to understand which information flows where), or arrows are drawn everywhere and the chart becomes unreadable.
- Gantt charts **focus on timing as the only constrained resource** and assume by assigning a row per task that human or capital resources are unlimited. A Gantt chart may show that two tasks can be performed in parallel, but if only one task can be performed at a time due to human or equipment limitations, they will have to be completed sequentially. This comes back to the concept of **critical chain** explained above.
- Gantt charts present a **deterministic schedule of a process**. There is no room for a probabilistic distribution of the duration of tasks, each task is assumed to have a definite duration whatever the circumstances, which may not be a very realistic assumption.
Gantt charts provide **poor management of iterations**. Although allowed, contrary to both previous tools, loops between concurrent tasks are hard to represent, or must be defined deterministically — contradictory to the typical loop pattern: try, test, modify, retry which only ends when the test results are considered to be satisfactory. A Gantt chart will have to specify the number of iterations by repeating a task box a definite number of times, as figure 2.1.1.7 shows.

![Gantt Chart](image)

**Figure 2.1.1.7: Some drawbacks of Gantt charts: iterations and links crossing over**

Figure 2.1.1.7 also shows how a Gantt chart can get quite disorganized when all relationships between tasks need to be shown. Although the example remains simple, Gantt charts can be difficult to understand when arrows begin to cross over and the visualization advantage of Gantt charts is lost. This example also demonstrates the rigidity of Gantt charts when representing iterations. Tasks A and B are looped and if the review at the end of B does not meet the targets then rework on A is necessary before rework on B, but not necessarily for the whole initial duration. Gantt charts base time scheduling on the assumption of a certain number of iterations, three in the example. However only the course of the project can determine if there will need to be one or several iterations.

While Gantt charts are easily created using Microsoft Excel, several software packages handle Gantt charts more easily. Microsoft Project is probably the most widely used software but less renowned software is also available, such as Smartdraw.

**D. The Design Structure Matrix (DSM)**

None of the above process mapping tools really fit the large-scale complex product development process requirements for this thesis, in particular iterations and uncertainty. Although Gantt charts are useful once the process is mapped to lay out the process scheduling, they are not well suited for mapping it in the first place. The method preferred by the author to model Product Development Processes, the Design Structure Matrix, will be fully described in Chapter 3.

**2.1.2 Incremental Process Improvement vs. Process Reengineering**

Companies put a lot of effort into improving their processes. Most of the time, it is through minor modifications ("incremental improvements" or "continuous improvements") that do not radically modify the habits of the employees while yielding some tangible efficiency improvement. After many years of incremental process improvements, companies face more
difficulties in finding incremental improvements opportunities, and they yield decreasing results. Michael Hammer and the Process Reengineering advocates argue that such incremental process redesigns will yield a few percents of efficiency improvement while a more radical approach, called Process Reengineering, has proven to yield results of efficiency improvements by 70% and more. Many real world implementations support such results. Details of the approach and case studies are very well described in (Hammer and Champy, 1993) and (Hammer and Stanton, 1995). The approach is also summarized in the papers (Hammer, 1990) and (Davenport, 1995). This section outlines differences between incremental improvement and process reengineering.

Process Reengineering consists in creating a brand new process to fulfill a function without taking into account any historical or traditional consideration of how the work has been done up to the reengineering effort. The idea is that current processes are inherited from long traditions and evolutions through incremental process improvements, which have in the end made them totally obsolete and irrelevant to the goal they pursue. A key metric that Michael Hammer uses constantly is the ratio between time spent on value-added work as opposed to total elapsed time throughout the process ("VT over ET" for "Value-added time over Elapsed time"). He claims that our processes are full of non-value-added tasks like coordination and pure waste (hand-offs while operators are working on something else) that pollute the process and cost time and money for nothing that the customer might value. Process Reengineering consists in centering processes on value-added tasks, tasks that bring value to the customer, and limiting non-value-added ones to their absolute minimum. Figure 2.1.2.1 gives an example of process reengineering at an insurance company, taken form Michael Hammer’s lecture notes of class 15.769 at MIT.

**Figure 2.1.2.1a: Functional organization claim resolution process: duration 7 to 10 days**

**Figure 2.1.2.1b: Reengineered claim resolution process**
All non-value-added tasks were eliminated from the claim resolution process. Going through the local agency was not adding any value, no work was done there, the claim was just forwarded to the claim manager. The claim manager was actually the only one doing work in the process apart from the adjuster inspecting the car, but his work was non-value-added, he simply would assign claims to adjusters. In the reengineered process, the claim is randomly assigned to the first adjuster that picks up the phone, who then locates the closest inspector to the car accident location. The waiting time between the car accident and the car inspection was thus reduced by a factor of around 24... Increasing the productivity of employees in the old process would never have achieved such high results.

Table 2.1.2.2 exhibits major differences between incremental process improvement versus process reengineering.

<table>
<thead>
<tr>
<th>Incremental Process Improvement</th>
<th>Process Reengineering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relatively easy implementation</td>
<td>Resistance of functional teams and workers</td>
</tr>
<tr>
<td>Limited results</td>
<td>Dramatic efficiency improvements</td>
</tr>
<tr>
<td>Low risk of failure</td>
<td>High: risk of failure</td>
</tr>
<tr>
<td>No need for high-level management support</td>
<td>Need for top management support</td>
</tr>
<tr>
<td>Functional over-segmentation of work remains</td>
<td>Process dominates over functions</td>
</tr>
<tr>
<td>Little organizational change required</td>
<td>Organization structure must change radically</td>
</tr>
</tbody>
</table>

Table 2.1.2.2: Major differences between incremental process improvement and process reengineering

This thesis actually fits in between the two categories. The process analysis and improvement method using the Design Structure Matrix proposed in the following chapters surely does not fit in the description of Process Reengineering: it does not put forward company wide transformations that would require top management support. It does not radically change the organizational structure, although it usually proposed some important changes in teamwork enablers like project-wide individual performance evaluation criteria alignment for example. However, the reader will see that DSM-based product development process analysis and improvement does more than incremental process improvement as presented in table 2.1.2.2, essentially by promoting the process-wide vision that process reengineering gives through the documented DSM. This discussion will become clearer when the case of Chapter 5 is presented.

2.1.3 Product Development Process Design to Manage Risks
To finish this section about process mapping and (re)design, a methodology recently put forward by Dr. Darian W. Unger, with whom the author enjoyed sharing an office at MIT for two years, was worth mentioning because it applies specifically to product development. In his Ph.D. dissertation (Unger, 2003), Dr. Unger first reviews and identifies many different types of Product Development Processes by looking at metrics on iterations (the breadth of iterations, the number of inter-phase loops, the level of planning) and on the reviews that mark the process (rigidity of the reviews, frequency of the reviews).

These metrics are then used to see how the product development process characteristics fit the risks that the company faces (technical risk of not meeting requirements, market risk of not designing a successful product, schedule risk of not meeting important deadlines, budget risk of
exceeding planned expenses and other risks including organizational or regulatory). It turns out that all ten case studies fall between two extreme product development processes, the “stage gate” process with few, rather unplanned and narrow iterations and frequent and rigid reviews, and the “spiral” process with many planned and comprehensive iterations and less frequent and less rigid reviews. The stage gate emphasizes control and predictability, primarily to address technical risk and is wide-spread in manufacturing companies, as opposed to the spiral process that allows flexibility and adaptability during the process to evolving market conditions, hence lowering market risk.

A product development process procedure is finally proposed, following the steps below:

1. Identify and prioritize project development risks
2. Assign each risk to a specific phase or cycle
3. Plan the necessary iteration cycles to address the assigned risks
4. Schedule reviews at the completion of key phases or cycles

Application of this procedure to a manufacturing company to realign its product development process with the risks it faces offers promising results.

2.2 Organizations Management

This field is so vast that is would be useless to try to cover it comprehensively in this section. (Ancona et al., 1999) is a good reference book on all issues that Organizations Management can comprise: organizational structure analysis (through three lenses: strategic, political, cultural), efficient conduction of team projects, the diversity of cognitive styles in teams, team processes (decision-making, etc.), linking teams to their organizations, workforce management (employment relationships), the management of change in organizations, international aspects of organizational processes, managing cultural diversity, negotiation and conflict resolution, etc.

The intent of the author is solely to provide the reader with the basics of organization structures across a large number of companies. Three main strategies in structuring an organization prevail: grouping, linking and aligning.

Grouping gathers together some tasks, functions or disciplines, and separates them from others. Grouping by activity leads to what is known as Functional Organizations, Engineering being separated from Manufacturing and Marketing & Sales, for example. This type of structure allows each function to deepen its specialization and achieve efficiencies through scale economies. However, these advantages come at the cost of integration across activities, individuals tending to develop narrower perspectives and have difficulties in solving problems that require joint efforts with other groups.

Grouping by Output organizes on the basis of the service or product provided: in this design, functions are divided across the business or product line they support. If a company follows this pattern and divides into business A (e.g. industrial products) and business B (e.g. consumer products), each business will have engineering, manufacturing, marketing and other functions within the business division.

Grouping can also be done by geographic region.
Yet, fewer and fewer companies are finding that grouping an any single dimension (activity, output, geography) is adequate. Corporate strategies frequently require attention to multiple priorities simultaneously, product and function, for example. Many organizations have turned to multifocused grouping of structures in an attempt to break out of the constraints imposed by a single mode of strategic grouping, and to build a matrix structure. In a business / functional matrix, for example, the traditional functional departments cross the product line divisions, so that functional supervisors within the product line report to two bosses: the functional manager and the product line manager. The organization benefits from the information exchange and control provided by the grouping of people by function but also achieves integration across product-related activities.

The differences between functional organizations and product / functional matrix structures will be helpful to review observations on organizational issues made during the case studies.

2.3 Project Management

Product Development specific Project Management is well described in (Ulrich and Eppinger, 1995, 2000). It is mostly a mix and an implementation of the ideas and tools presented in the previous two sections. It relies primarily on an understanding of the tasks to be achieved and their relationships, which was largely covered in the Process Mapping section 2.1. Staffing and organizing the project can benefit greatly from organizational managements findings on team processes (small teams, full time assignment, cross-functional representation, etc.). Scheduling also largely depends on how precise and accurate the process map was: using Gantt charts presented in section 2.1 and / or DSMs presented in the next chapter will help build a realistic but optimized schedule. Budgeting is naturally key too. Since timing and cost are very uncertain at the beginning of a project, some important margin should be added to the budget. Identifying project risk areas and assessing the risks right from the beginning can also help prevent critical unexpected developments of the project. Finally, many practical matters such as information sharing (through information systems, meetings or public display for example) will have a significant impact on the success of a project. Project managers need to be aware of all these important factors to enable creativity, productivity and success.

2.4 Knowledge Management

Knowledge Management is a broad term that encompasses notions or popular terms like organizational learning, intellectual capital management, or intellectual asset management. Several books detail the philosophy of enabling knowledge generation, development, capture and transmission throughout the enterprise and describe hundreds of cases. (Ahmed et al., 2002) offers a good mix of both theoretical work and case study analysis. It explains how successful knowledge management programs highlighted numerous potential benefits, such as:

- Improved innovation leading to improved products and services
- Improved decision making
- Quicker problem solving and fewer mistakes
- Reduced product development time
- Improved customer service and satisfaction
- Reduced Research and Development costs.
Literature Review

The specific area of knowledge management that is mainly of interest for this thesis is system-level knowledge, that is knowledge that enables easier and more efficient systems engineering. System-level knowledge is usually very poorly documented and companies rely on their experienced system engineers to accumulate and reuse system-level knowledge. Dr. Qi Dong found in her M.S. Thesis (Dong, 1998, p.117) that only 30% of the system-level knowledge was usually recorded in documents, as opposed to 70% found through interviews with engineers. Recording system-level knowledge is one of the purposes of this thesis, and the next chapter, which presents the Design Structure Matrix (DSM) methodology, will help reach that goal. Much of the DSM literature deals with knowledge management indeed.

2.5 Chapter Summary

Reviewing important literature in this chapter has helped introduce some of the key concepts at stake in this thesis. The notion of process thinking and appropriate tools to do so were described in detail before less comprehensive introductions of organizations management, project management and knowledge management issues relevant to this thesis were accomplished. The reader is now ready to jump to the work at the heart of this research: the Design Structure Matrix methodology in chapter 3, necessary to understand the rationales and conclusions of the case studies in chapters 4 and 5.
Part II: Methodology & Applications
3 METHODOLOGY – THE DESIGN STRUCTURE MATRIX

This chapter briefly introduces the method that is used throughout the thesis: the Design Structure Matrix. The information presented here is based on the Design Structure Matrix website at MIT (web.mit.edu/dsm) and on previous publications. The objective is to provide the reader with a broad understanding of the method and its applications. However, for a complete and detailed journey on DSMs, the reader is invited to consult the references mentioned.

3.1 The DSM basics

The Design Structure Matrix (DSM) is a tool that helps capture the important relationships in a complex system. By focusing on interactions and information flows, the DSM allows to analyze the structure of systems and thereby to organize and manage them more efficiently. First introduced some twenty years ago (Steward, 1981a), the DSM method truly received attention in as MIT, among others, used and developed it to model design processes (Eppinger et al., 1994). The method consists in recording in a square matrix all the interactions inside a system and in deducing a better organization from these interactions. The different types of systems and interactions that can be recorded will be detailed in section 3.2. Let us start with a simple abstract example. The left part of Figure 3.1.1 shows a system of 9 elements, for the purpose of this thesis, let us suppose that the elements represent the tasks that need to be completed in a design process. These tasks are randomly numbered in the chart, and they influence each other through the pattern of arrows. It is quite difficult to understand and visualize the structure of the example through the diagram in the left part. On the right part of figure 3.1.1 is the corresponding DSM that maps the interactions between process tasks, listed in rows and columns in the same order. The matrix is filled with oriented interactions from the elements in columns to the elements in rows. Thus, looking at the off-diagonal marks in a column, task 7 in the example of figure 3.1.1, will describe which other tasks depend on the output of the task corresponding to the column, task 4 in the example. Similarly, the off-diagonal marks within a row, 5 in the example, represent all of the tasks whose output is required to perform the task corresponding to that row, i.e. 1, 3 and 8 in the example. The diagonal cells are usually meaningless since they would represent a dependency of a task on itself.
Figure 3.1.1: How DSMs are constructed and read, a simple example

From the way the matrix is constructed, if the design process modeled is achieved in the order of the elements in rows and columns, the triangle below the diagonal represents feed forwards (information flows of low rank to a higher rank in the sequential design process), and the triangle above the diagonal represents feedbacks (information flows that go in the reverse order of the design process). See figure 3.1.2 for a graphical illustration of these two distinct sets of information flows.

Figure 3.1.2: Feed forwards and feedbacks in a DSM

Feedbacks can be disturbing because they involve critical assumptions and probable rework, iterations, in the design process. In the example above, task 1 requires information from tasks 5 and 9 which have not been achieved yet, so assumptions on what information they are likely to come out with is necessary to initiate the process with task 1. Furthermore, when the process goes on and reaches task 5, it is probable that the estimations made earlier on its outcomes are not acceptable anymore now that the process has actually reached task 5. Rework on task 1 might hence be necessary, and the process can start again. Such iterative loops can really be
time-consuming, involve important delays in the initial plan if they were not predicted, as well as unexpected cost if major investments are made through the process and need to be adapted later.

**However, not all iterations are undesirable.** The classic loop design – test – redesign is for example usually truly value-adding because the complexity of nowadays’ systems does not allow product developers to fully trust their design without testing it; tests tell them what the issues are and redesign is necessary to fix the problems and improve product quality. Such a loop is therefore necessary and actually desirable.

Sources of feedbacks and iterations can be:
- Results of test that are necessarily done later
- Results of planned design reviews
- Design mistakes
- The natural pattern of the internal constraints of the system being designed.

By reorganizing the process (reordering rows and columns), DSMs make the marks above the diagonal migrate below the diagonal and **identify which iterations are necessary, and which simply result from a non-optimal sequence in the design tasks.** Such reorganization is performed in a **two step method.**

1. **Partitioning:** it consists in decomposing the process in clusters of tasks that are interdependent. It is a purely mathematical algorithm and does not involve any knowledge on the particular process. Several software packages perform this step and other things, for example PSM32 by Problematics (Don Steward’s company), DeMAID/GA (Design Manager’s Aid for Intelligent Decomposition with a Genetic Algorithm) or a Microsoft Excel Visual Basic add-in programmed at MIT and available on the aforementioned website. More precisely in mathematical terms, what the algorithm does is change the order of the elements in rows and columns so as to reduce the matrix to a block triangular one, if possible. See the result of partitioning for the example of previous figures in figure 3.1.3.

2. **Optimizing Cycles:** this second step deals with reorganization inside the cycles identified during partitioning. Because they are cycles, some feedback is unavoidable and necessary inside of them, and each cycle will represent a group of tasks that will have to be integrated in a common team. Using knowledge on the particular process studied, this step consists in selecting the information flows that are most acceptable as feedbacks (that is those for which assumptions on the outcome are easy to make, or for which the intensity of the influence on other elements is lowest). Several methods can be used in this step (“tearing” is one of them¹ - see Steward, 1981b or the DSM website at MIT). Because it involves knowledge on the particular process studied, there is no mathematically unique resulting DSM. For the example given so far, several ones might be acceptable depending on the real process modeled. One is shown in figure 3.1.4.

Nevertheless, it can happen that DSMs of highly coupled systems simply cannot be partitioned, they just do not meet the mathematical requirements to be decomposed so. In that case, manual

---

¹ Tearing is the process of choosing the set of feedback marks that if removed from the matrix (and then the matrix is re-partitioned) will render the matrix lower triangular. The marks that we remove from the matrix are called "tears". Identifying those "tears" that result in a lower triangular matrix means that we have identified the set of assumptions that need to be made in order to start design process iterations when coupled tasks are encountered in the process. Having made these assumptions, no additional estimates need to be made.
clustering, that is manual reordering of the elements in rows and columns, is the only way to reach a matrix as close to block triangular as possible.

![Partitioned DSM](image1)

**Figure 3.1.3: Partitioned DSM**
It is block triangular: there are no feedbacks outside the diagonal blocks

![One possible reorganized DSM](image2)

**Figure 3.1.4: One possible reorganized DSM**
Tasks 5 and 8 were permuted so as to finally keep two acceptable feedbacks only

Figure 3.1.5 shows the possible task interaction patterns that a reorganized DSM can exhibit.

![Task interaction patterns in a reorganized DSM](image3)

**Figure 3.1.5: Task interaction patterns in a reorganized DSM**
Source: Dr. Daniel E. Whitney

Also, DSM analysis is not limited to resequencing tasks or regrouping elements in clusters, one step beyond is to modify the system (if possible and appropriate: by redefining existing tasks, splitting existing tasks into two or more pieces, creating completely new tasks or deleting old tasks) to allow breaking cycles and reducing development time. Yassine, Whitney and others (Yassine et al., 2000) have shown the advantages of such an approach through real world cases at Ford Motor Company (hood and seat belt design).
Methodology – The Design Structure Matrix

It can easily be seen on the examples above how powerful DSMs can be to base the organization of processes on interactions. It is not their only benefit:

- Thanks to the software packages that handle partitioning automatically, DSMs help structure large systems as well as small ones indifferently, contrary to PERT, IDEF3 & Gantt charts.
- DSMs help visualize the intrinsic complexity and structure of a system or process because they capture all interrelationships, including feedback loops and cycles, which other mapping tools do not allow.
- DSMs record on a single document system-level knowledge, i.e. knowledge on the structure of the system and the complex interrelationships between its elements. DSMs therefore ensure transmission of this knowledge, usually not documented and only gained after many years of experience, to inexperienced people. It provides them easier and faster learning of the complexity of a system.
- DSMs approach the organization of design processes on the basis of the actual exchanges of information in the process, not on an a priori, physical, structural, cultural or historical inherited decomposition of it.
- DSMs remove unnecessary design loops and rework from processes, and identify the causes of unavoidable ones, which is a good starting point for further effective process modification and improvement (see Yassine & al, 2000).
- DSMs make a very useful project management tool. Unlike other tools, e.g. PERT charts, they comprise iterative loops and therefore ensure more accurate time and cost planning (if lead time and cost are assigned to each task). DSMs can also help trace the consequences of a modification on a part of the system through the downstream sequence of information flows.

The practical way the DSM method is implemented to study a system or process involves several steps:

1. Define the system and its boundaries
2. List all the system elements or process tasks
3. Study the information flows between the elements or tasks (through requirements documents and interviews with experienced engineers). It has proven more efficient and more accurate to ask interviewees about their inputs or what influences their system elements rather than what information flows out of their elements. Since they depend crucially on them, people have indeed a better understanding of what their inputs are than how their elements influence others. It is therefore recommended to fill DSMs following rows rather than columns.
4. Build a matrix to represent the information flows, verify it with engineers
5. Partition the matrix
6. Optimize inside cycles using knowledge on the system or process
7. Give the matrix to the engineers and managers
8. Consider a reorganization of the process and eventually implement it.

The DSM building process is very iterative itself: often times, as the DSM practitioner goes through requirements documents and interviews engineers, boundaries of the system are expanded, new elements need to be added and data collection on interactions from and to these new elements has to start again.
This is one of the drawbacks of the DSM method. Here is a more detailed set of issues that derive directly from the DSM method implementation process.

- The system or process studied must already be well known by engineers before the study can be performed because the interviews would otherwise not lead to a complete understanding of the system interactions. As a consequence, totally innovative systems for which there is no previous experience cannot be studied through the DSM method as presented above. Section 3.3 will present a potential to this issue.

- A DSM project is very time consuming, especially the interview process, both for the analyst conducting the study and the engineers participating to the interviews. Any reasonably limited system will require several weeks full time to be thoroughly studied.

### 3.2 Different types of DSMs and their applications

This section reviews the different types of interactions that can be modeled in a DSM and the corresponding applications and outcomes. Table 3.2 summarizes the four popular categories of DSMs. As shown in the table, depending on the nature of elements listed in rows and columns (DSM data types columns of table 3.2) the application and analysis method will change. When analyzing the structure of a system based on the interactions between its components or subsystems, the notion of sequence is not very important. What comes out of such an analysis is an idea of the groups of components that share an important interface and that need to be designed with careful cross-component interface management focus. The nature of the relations recorded can also vary within each category of DSM: for component-based DSMs, the relationship studied can be spatial (physical interface), functional or appearance (Dong, 1998).

<table>
<thead>
<tr>
<th>DSM Data Types</th>
<th>Representation</th>
<th>Application</th>
<th>Analysis Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component-based</td>
<td>Multi-component relationships</td>
<td>Systems architecting, engineering and design</td>
<td>Clustering</td>
</tr>
<tr>
<td>Team-based (organization DSM)</td>
<td>Multi-team interface characteristics</td>
<td>Organizational design, interface management, team integration</td>
<td>Clustering</td>
</tr>
<tr>
<td>Activity or task-based (→ process mapping)</td>
<td>Activity input / output relationships</td>
<td>Project scheduling, activity sequencing, cycle time reduction, critical path analysis</td>
<td>Sequencing &amp; partitioning</td>
</tr>
<tr>
<td>Parameter-based</td>
<td>Parameter decision points and necessary precedents</td>
<td>Low level activity sequencing and process construction</td>
<td>Sequencing &amp; partitioning</td>
</tr>
</tbody>
</table>

**Table 3.2: Summary of the four popular categories of DSMs**

Source: MIT DSM website web.mit.edu/dsm

Similarly, team-based DSMs identify teams that exchange a lot of information and whose cooperation therefore needs to be enabled by the organizational structure of the company. The notion of sequence does not apply to teams, they are simply clustered together. Activity or task-based DSMs apply to studying work processes, and the reorganized DSM’s key outcome is an optimal sequence of tasks that minimizes unnecessary iteration. Parameter-based DSMs also
Methodology – The Design Structure Matrix

look at design process construction, but from the point of view of the activity of freezing a parameter before another.

Examples of applications are numerous. Although most of them use more advanced approaches to be presented in the next section, the type of DSM they use is worth mentioning. The most relevant are:

- (Eppinger et al., 1994) presents a parameter-based DSM for car brake system design and a task-based DSM of the design process of semiconductors.
- (Pimmler & Eppinger, 1994) presents the architectural analysis (component-based DSM) of climate control systems at Ford Motor Company
- (Dong, 1998) studied vehicle door design and throttle body system interface and assembly design at Ford Motor Company using a mix of component-based and parameter-based DSMs
- (Browning and Eppinger, 1998) use a task-based DSM to study and simulate the design process of an aircraft
- (Sosa, Eppinger and Rowles, 2001) use a component-based DSM and a team-based DSM to compare design and organizational structures for a large commercial aircraft engine.

Chapter 4 will use a parameter-based DSM to study a subsystem of an engine at PSA Peugeot-Citroën and the structural change induced by an important innovation. Chapter 5 will use a variant of a task-based DSM (deliverable DSM, with a completion date embedded in each activity as opposed to just the nature of the task) and organizations DSM to study the design process of a large subsystem of vehicles at Ford Motor Company.

The DSMs presented so far are merely binary: the cell intersecting a row and column is either empty or marked with an X or a 1. A lot more applications rely on numerical data entered in the cells rather than simply binary marks. Section 3.3 reviews some more advanced possibilities of analysis using numerical DSMs.

3.3 Numerical DSMs and other advanced applications

The value of the DSM method is further illustrated in applications using values in the matrix rather than binary marks. Examples of applications follow.

- **Level numbers**: Steward (1981) suggested the use of level numbers instead of a simple "X" mark, for certain marks in the binary matrix. Level numbers reflect the order in which the feedback marks should be torn. The mark with the highest level number will be torn first and the matrix is reordered (i.e partitioned) again. This process is repeated until all feedback marks disappear. Level numbers range from 1 to 9 depending on the engineers judgement of where a good estimate, for a missing information piece, can be made.

- **Dependency strength**: information flows can be prioritized depending on the intensity of the relationship described. The partitioning algorithm will then minimize the sum of dependency strengths above the diagonal.

- Similarly, **the volume of information exchanged** can be a measure of the dependency strength.
• **Variability of information exchanged** (task-based DSMs): A variability measure can be devised to reflect the uncertainty in the information exchanged between tasks. This measure can be the statistical variance of outputs for that task accumulated from previous executions of the task (or a similar one). However, if we lack such historical data, a subjective measure can be devised to reflect this variability in task outputs.

• **Probability of repetition** (task-based DSMs): this number reflects the probability of one activity causing rework in another. Upper-diagonal elements represent the probability of having to loop back (i.e. iteration) to earlier (upstream) activities after a downstream activity was performed. While lower-diagonal elements can represent the probability of a second-order rework following an iteration (Browning and Eppinger, 1998). Partitioning algorithms can be devised to order the tasks in this DSM such that the probability of iteration or the project duration is minimized. (Browning, 1998) devised a simulation algorithm to perform such a task, which was applied by (Yassine et al., 2001)

• **Impact strength** (task-based DSMs): this can be visualized as the fraction of the original work that has to be repeated should an iteration occur (Browning, 1998). This measure is usually utilized in conjunction with the probability of repetition measure, above, to simulate the effect of iterations on project duration.

• **Interaction characteristics**: instead of a number or a mark, the nature of the information flow can be specified in the DSM cell (Pimmiller and Eppinger, 1994 have for example 4 different types of interaction recorded in their Climate Control System component-based DSM: spatial interaction, information, energy and materials flows). For Team-based DSMs, such characterization can apply to the level of detail of the information exchange, the frequency, the timing, etc.

Other research and applications include deriving a DSM for requirements documents in order to be able to build DSMs of innovative systems that have never been designed before. The DSM building process described earlier relies indeed on the experience of engineers interviewed, which is missing when the system is new. Some promising research on that subject was completed in 2002 by Dr. Qi Dong in her Doctor of Philosophy in Mechanical Engineering Thesis entitled “Predicting and Managing System Interactions at early phase of the product development process” (Dong, 2002). It proposes to use the Axiomatic Design theory and framework to relate critical design parameter with functions in a Design Matrix derived from the specifications documents. It then exhibits a method to convert the Design Matrices into DSMs.

### 3.4 The “Whitney ratio”

A very interesting finding of DSM research is a consistent average number of marks per row (also known as the “Whitney ratio”, in the name of Dr. Daniel E. Whitney who calculated and discovered such consistency) of 6.2 across numerous DSM projects of different scope in different industries. Figure 3.4 shows the regression line and the correlation seems to be very sound.

This finding is somewhat mysterious because it implies that there is an underlying structure of many human systems such that internal interactions show a pattern of an average of 6.2 connections per element.
Figure 3.4: Consistent average of 6.2 number of marks per row in numerous DSM studies

One first possible qualification of this result is that simpler systems with less intense interaction pattern (lower average number of marks per row) are simply not studied with DSMs because there is no need for such a powerful tool to understand their structure. Or that such studies if they exist are not published. An interpretation could therefore be that systems created by humans, at least in the engineering world (the only domain represented in figure 3.4, but ranging from cars to construction through aircrafts, hospitals and pharmaceutical facilities), have no more than around 6.2 average number of connections per element. This could be a discovery of an important limitation of the human brain. However, the question remains very open and no brain & cognitive science experiment has been conducted to verify such a hypothesis.

Nevertheless, even without being able to interpret fully the result of figure 3.4, the Whitney ratio remains an interesting metric to keep in mind when working with DSMs. Chapters 4 and 5 mention the observed Whitney ratios in the DSMs resulting from their corresponding case studies, and try to interpret any difference with the 6.2 average.

3.5 Chapter Summary

This chapter briefly gave an overview of the Design Structure Methodology, its various applications and value, which will be extremely useful for the reader to follow the case studies presented in Chapter 4 (a classical application of the DSM methodology to studying the structure of a subsystem of car engines at PSA Peugeot-Citroën in Paris, France) and Chapter 5 (a more original and more in-depth study of the design process of a large subsystem of cars at Ford Motor Company in Dearborn MI). These applications will further support the ideas presented in this chapter and their value in the Product Development World.
4 APPLICATIONS: HYBRID VEHICLES AND ACCESSORY DRIVE SYSTEMS AT PSA PEUGEOT-CITROËN

This chapter presents some applications of the concepts and methods presented in the previous chapters to a real situation in the PSA Peugeot-Citroën group. The impact of technological innovation on the structure of the Front End Accessory Drive (FEAD) System using DSMs and Knowledge Management practices were studied in January 2002 in the surroundings of Paris, France. The complexity of FEAD systems is too limited to make this case study fit in the objectives that this thesis aims at (i.e. the concurrent mapping of processes and management of organizations, knowledge and projects using a single approach in large-scale design projects). However, the chapter will introduce the reader to real world applications of the methodology and key concepts presented earlier.

4.1 Background

The author’s contacts with PSA Peugeot-Citroën dated back to a one month internship on the final assembly line of the plant in Wuhan, China, in summer 2000. Later on, on the occasion of a career fair, the author met Mr. Jean Martin Folz, Chairman and CEO of PSA Peugeot Citroën. Interested by the research conducted at the MIT Center for Innovation in Product Development, he directed the author, with the help of Mr. Jean-Louis Grégoire, Senior Manager for Executive Development (Directeur de la Politique Cadres), to people working on similar subjects in the Direction of Research and Automotive Innovation (Direction de la Recherche et Innovation Automobile, DRIA) of PSA. That’s how a one month case study in January 2002 was set up.

The objectives of the case study were to:
- Present the DSM method to PSA
- Exchange views on Knowledge Management
- Learn from other methods used at PSA, compare them with the DSM method
- Apply and experience the DSM method on a practical case with potential reorganization perspectives
4.1.1 Design process at PSA
Reviewing the general design process at PSA Peugeot-Citroën is useful before getting into the detail of the system chosen for the study.
The car design process at PSA is organized in two major subsequent phases, l’Avance de phase (Phase advance), non vehicle specific, handled by the Direction of Research and Automotive Innovation (DRIA), and then Development, vehicle specific, handled by the Direction of Technical Platforms and Purchasing (DPTA). [For a summary of the PSA group organization chart, please refer to Appendix A.] Each large phase divides into two subsequent steps:
1. Phase advance
   • Generic phase advance (l’avance de phase générique), where many innovative technologies and concepts are studied, tested and developed for themselves, independently of a particular vehicle. If they prove to work and are useful, they might be included in the next step,
   • Programmed phase advance (l’avance de phase programmée), which studies and selects different vehicle concepts.
2. Development
   • Preliminary Definition, which defines the vehicle’s composition, the necessary elements to build a technically and economically coherent vehicle.
   • Detailed Development itself, where all the parts are designed and prototypes are tested.

![Diagram of design process]

**Figure 4.1.1 : General organization of the design of a car at PSA Peugeot-Citroën**

4.1.2 Choice of the System studied
The choice of an appropriate system for the case study was constrained by several factors and involved delicate tradeoffs.
• The system had to be simple enough to be comprehensively studied within four weeks.
• It also had to be complex enough to permit non trivial results that would be of interest both for PSA and research.
• The system had to be already fully designed or at least in the last phase of design (see the constraints of the DSM method in chapter 3).
• In order to lead to practically usable conclusions for PSA in the future, the system also needed to be subject to current or further redesign later
• The people involved in the design of the system needed to be easily reachable and reasonably available for interviews
Applications: Hybrid Vehicles and Accessory Drive Systems at PSA Peugeot-Citroën

From the very beginning, Catherine Boulanger, the internship supervisor, directed the author to the hybrid vehicles projects of PSA because hybrid vehicles constitute the short and middle term future of the automobile industry and because teams involved in those projects had already extensively used project management tools in the past. That meant that they had many documents available, and an interesting method comparison perspective was expected.

The Front End Accessory Drive system (FEAD) was chosen, because it held an important interface role between the engine and the accessories, because its design was deeply challenged by new hybrid specific requirements as compared to accessory drives in regular thermal vehicles, and because it still remained a fairly simple system.

4.1.3 Analysis objectives
As said earlier, the main objective of the case study was to analyze the organization of a particular system design process using DSMs. When the FEAD was chosen to be the support for this case study, several goals were identified.

• Analyze the current design process for the system in conventional thermal propulsion vehicles, see how the DSM method could help reorganize it
• Analyze the design process for the system in future hybrid vehicles, in which several more constraints are imposed to the FEAD, see how the DSM method could help organize it
• Compare both processes and find out what the impact of innovation from the conventional thermal one to the hybrid one was on the organization of the process.

The internship therefore took place both in the detailed development phase for the thermal engine FEAD systems design, and in the generic phase advance, the hybrid FEAD system being developed in general, with no particular vehicle or market segment target defined yet.

4.1.4 General description of the conventional thermal FEAD system
The Front End Accessory Drive system, further referred to as FEAD, is intended to drive the accessories by transmitting to them the torque coming from the crankshaft. On some engines, some accessories are driven by a Rear End Accessory Drive, mounted on one of the cam shafts, but this system was not considered for this project.

The generic term “accessory” can incorporate
• the alternator
• the air conditioning compressor
• the power steering pump
• other pumps like water pump (cooling system) or vacuum pump

The main issues that have to be addressed in the design of FEAD systems are:
• Different rotation regimes for the engine and the accessories (therefore different diameters of pulleys are required)
• Sliding must be avoided (large winding angles of the belt around pulleys are aimed at)
• Structural integrity and reliability
• Constraining space requirements
• Noise reduction
• Ease of maintenance and disassembly
FEADs include the following elements (see figures 4.1.4.1 and 4.1.4.2):
- the belt itself
- the dynamic tensioner (which ensures that the soft leg of the belt remains tightened)
- idlers (which ensure sufficient winding angles and keep free legs of the belt short)
- the crankshaft pulley and that of the accessories

![Figure 4.1.4.1: Picture of a FEAD](image1)

![Figure 4.1.4.2: Scheme of the FEAD](image2)

The general tendency is to separate the power steering pump from the rest of the FEAD and to operate it with an electrical motor of its own. Reasons for this trend is to decouple the intensity of the power steering function from the engine speed as well as simplify the path of the belt and potentially to get rid of an idler. Consistently with this evolution and so as to simplify the analysis, the following general framework was chosen:

![Figure 4.1.4.3: Framework chosen for the DSM analysis](image3)
Two other features need to be specified before moving on to the comparison with the Mild Hybrid system.

- The crankshaft pulley holds a special feature, the Torsion Vibration Damper (Amortisseur de Vibration de Torsion, AVT), whose role is to decouple the torsion resonance frequency and to filter the acyclisms of the engine (i.e. the variations of torque due to the discontinuous explosive nature of 4-stroke engines). AVTs consist of one (simple AVTs) or two (double AVTs) circular rubber layers and a heavy inertia ring embedded in the crankshaft pulley. Double AVTs are mainly used with diesel engines because they exhibit more acyclism than gas engines. To simplify the analysis, only simple AVTs were included in the studied system.

- The alternator and the engine have a different inertia. Therefore, at every variation of engine speed, the alternator does not adapt its rotation speed at the same pace and it hurts the belt, especially during decelerations. To cope with this problem, engineers have installed a ball bearing one-way free-wheel system on the alternator pulley, that lets the alternator decelerate at its pace but that drives it normally while accelerating. This free-wheel system was incorporated in the base system studied.

### 4.1.5 Additional functions and constraints for the hybrid version

It is useful at this point to briefly describe the architecture of the Mild Hybrid Vehicle concept. Mild Hybrid Vehicles are and will be among the very first hybrid vehicles to be sold on the market. They start from a conventional thermal propulsion car, and “simply” replace the alternator with a reversible electrical machine that can be used both as an alternator and as a motor. Such replacement does induce major design issues, however, the general structure of transmission and powertrain remain unchanged. As a consequence, mild hybrid vehicles cannot be propelled by electricity only.

![Figure 4.1.5.1: Architecture of conventional thermal vehicles](image)

![Figure 4.1.5.2: Architecture of Mild Hybrid vehicles](image)

As the previous diagrams show, the FEAD is significantly affected by the switch from a purely thermal propulsion vehicle to a Mild Hybrid one, because it constitutes the interface between the engine and the electrical machine.
In addition to the regular function of driving the accessories by transmitting the crankshaft torque, the system inherits new functionalities represented in figures 4.1.5.3 to 4.1.5.6:

- **Start the thermal engine thanks to the torque of the electrical machine used as a motor** ("stop & start", enables to stop the thermal engine automatically at red lights and to restart it very quickly). In order to do so, one of the proposed features is a clutch on the crankshaft pulley that will make it possible to decouple the engine from the belt drive system and let the electrical machine run at high speed before the connection through the clutch is made and the engine is driven thanks to inertia (inertial start). Another option is to use epicycoidal gears that provide two different driving ratios depending on which drives what, the engine or the electrical machine. This last option was not considered during the analysis and the clutch system was adopted as the baseline for Mild Hybrid FEAD in the rest of the study.

- **Assist the thermal engine with additional torque provided by the electrical machine (boost)**, to improve the engine performance at low regime.

- **Drive the accessories by the electrical machine only** (thermal engine decoupled from the system thanks to the clutch).

---

**Figure 4.1.5.3**: Conventional mode

**Figure 4.1.5.4**: Boost mode

**Figure 4.1.5.5**: Inertial engine start mode

**Figure 4.1.5.6**: Purely electrical accessories drive mode
It can easily be seen on the figures above that some important issues arise from the variability of the drive modes.

- The driving ratios need to be adjusted to help the electrical machine start the engine, boost it and/or drive the accessories. In particular, the torque required to start the engine with the electrical machine is much higher than the regular torque transmitted by the belt in conventional mode (around 120 vs. 50 Nm).

- The fact that the electrical machine becomes the driving force swaps tightened and soft legs of the belt. As a consequence, the tensioner has to adapt to those variations of tension. That’s why it becomes a dynamic tensioner, mounted on a spring whose parameters will be chosen to address in the most robust way the different modes as well as the extension of the belt as it ages. (tensioners in thermal cars can also be dynamic for that last reason, but with different characteristics anyway)

- Still, the belt undergoes much more stress than in a conventional car, and must hence be strengthened.

- The crankshaft pulley, like all other pulleys, also suffers more torque and variation of torque than usual. Experimentations have shown that double AVTs would not resist those new conditions. Mild Hybrid vehicles are hence restricted to simple AVTs at this point.

- The free-wheel system included in the alternator pulley of conventional FEAD cannot be kept in mild hybrid systems because the alternator is required to drive the system itself: the one-way decoupling ball bearing is not compatible with this function. This feature was hence removed from the studied system for the hybrid version.

4.2 DSM analysis

4.2.1 Analysis framework

The type of DSM needed to be chosen among design parameter, task, physical / structural and organizations DSMs.

It would have been interesting to build a task DSM to compare its results with the OTT model, a Knowledge Management framework deployed through PSA (see a full description of OTTs in section 4.3): OTTs do not include feedback interactions between tasks and the comparison with a task DSM would have helped quantify how much information was missed by OTTs. Moreover, without feedbacks, OTTs cannot induce the task clusters that DSMs help identify. Nevertheless, the OTT building process was not completed yet for the FEAD when the case study started, and the author became aware of the existence of the OTT framework too late in the internship anyway.

As for organizations, the process is not really structured this way at PSA since no particular person is entrusted with a particular aspect of the system. Two kinds of designers participate in the process, project-related ones and cross-project experts. They must split the work some way during meetings, but that was not clear at all.

Physical connections and interactions did not seem to be so important in the system, as compared to dynamics issues for example.

Design parameter DSMs looked much more interesting, and it seemed easier to gather information on design parameter interactions in specifications documents. This is the type of DSM that was chosen.
Of course, designers did not have the habit to think in terms of interactions between the decisions on design parameters, but rather in terms of tasks to perform in order to design the FEAD, so the data gathering process was not easy. In particular, the adaptation to a design parameter approach was difficult because the set of design tasks does not match with the set of design parameters one by one, if at all. Nevertheless, after a few hours of work in common with experts on the subject, understanding of the method on their part and understanding of the system on my part helped work faster.

Since FEADs are almost never designed totally from scratch but rather incrementally modified from one engine to another, the current design process, especially in terms of design parameters, was not really formalized. Therefore, the DSM could not be initiated with a current organization order. An initial order had to be chosen: the design parameters were ordered by physical distribution.

The limited duration of the internship did not let the author get into sorting interactions by intensity. ‘1’s were put in cells that had an interaction and others were left empty.

The list of design parameters the experts and the authors identified are listed below. In order to grasp a complete understanding of the interactions within the system, several parameters of elements outside the system were included. Similarly, some input quantities for elements of the system, like the ambient temperature, have to be taken into account in the design. They were also added to the list, even though they are not proper design parameters.

Finally, the elements outside the system, including these overall input quantities, impose requirements to the FEAD system or to its parts. Since the system-level requirements were expected to be the core reason for the system to be coupled (because fulfillment of a system-level requirement requires a combination of different conditions on several design parameters), and even though they were not design parameters, it was necessary to include them in the analysis to build a representation faithful to reality. They were identified by “REQ” (requirements on the system imposed by the outside). Differences between conventional and hybrid vehicles appear in italics.

<table>
<thead>
<tr>
<th>Conventional Thermal FEAD</th>
<th>Mild Hybrid FEAD</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Belt</strong></td>
<td><strong>Belt</strong></td>
</tr>
<tr>
<td>Width, # of Vs</td>
<td>Width, # of Vs</td>
</tr>
<tr>
<td>Nominal length</td>
<td>Nominal length</td>
</tr>
<tr>
<td>Type (mechanical behaviour characteristics)</td>
<td>Type (mechanical behaviour characteristics)</td>
</tr>
<tr>
<td>Maximum Tension &amp; Tension Variation</td>
<td>Maximum Tension &amp; Tension Variation</td>
</tr>
<tr>
<td>REQ Mountability &amp; Accessibility</td>
<td>REQ Mountability &amp; Accessibility</td>
</tr>
<tr>
<td>REQ Sliding and grip</td>
<td>REQ Sliding and grip</td>
</tr>
<tr>
<td><strong>Dynamic Tensioner</strong></td>
<td><strong>Dynamic Tensioner</strong></td>
</tr>
<tr>
<td>Geometric amplitude</td>
<td>Geometric amplitude</td>
</tr>
<tr>
<td>Stiffness</td>
<td>Stiffness</td>
</tr>
<tr>
<td>Damper</td>
<td>Damper</td>
</tr>
<tr>
<td>Diameter</td>
<td>Diameter</td>
</tr>
<tr>
<td>Bearing (maximum acceptable side load)</td>
<td>Bearing (maximum acceptable side load)</td>
</tr>
<tr>
<td>Location</td>
<td>Location</td>
</tr>
</tbody>
</table>
Applications: Hybrid Vehicles and Accessory Drive Systems at PSA Peugeot-Citroën

**Idler**
- Diameter
- Bearing (maximum acceptable side load)
- Location

**Crankshaft Pulley**
- Diameter
- Inertia (of the pulley)
- Stiffness of the AVT rubber layer
- Inertia of the AVT inertia ring

**Engine**
- Inertia of the inner mobile system
- Min & max rotation speeds
- Standard user demand histogram
- Torsion resonance vs. speed curve, max torsion acceptable by the crankshaft
- Resisting torque when cold

**Alternator**
- Pulley diameter
- Inertia
- Torque absorption vs. speed curve
- Free-wheel system
- Rotation speeds
- Location

**Accessory (A/C compressor)**
- Torque absorption vs. speed curve
- Pulley diameter
- Inertia
- Bearing (maximum acceptable side load)
- Rotation regimes
- Location

**General FEAD System Requirements**
- REQ Space and safety distances
- REQ Noise, vibration and resonance
- REQ Reliability, life

**Environment data**
- Thermal ambiance
- Pollution ambiance (e.g. oil emissions)

**Total: 40 elements**

**Idler**
- Diameter
- Bearing (maximum acceptable side load)
- Location

**Crankshaft Pulley**
- Diameter
- Inertia (of the pulley)
- Stiffness of the AVT rubber layer
- Inertia of the AVT inertia ring
- Thickness (mainly impacted by the clutch)
- Maximum torque transmittable (clutch)

**Engine**
- Inertia of the inner mobile system
- Min & max rotation speeds
- Standard user demand histogram
- Torsion resonance vs. speed curve, max torsion acceptable by the crankshaft

**Reversible Electrical Machine**
- Pulley diameter
- Inertia
- Torque absorption vs. speed curve
- Torque production vs. speed curve
- Rotation speeds
- Location

**Accessory (A/C compressor)**
- Torque absorption vs. speed curve
- Pulley diameter
- Inertia
- Bearing (maximum acceptable side load)
- Rotation regimes
- Location

**General FEAD System Requirements**
- REQ Space and safety distances
- REQ Noise, vibration and resonance
- REQ Reliability, life

**Environment data**
- Thermal ambiance
- Pollution ambiance (e.g. oil emissions)

**Total: 43 elements**

The DSM building process presented in Chapter 3 was literally followed to produce the matrices that follow. As usual in DSM analyses, only "active" relationships are shown in the matrix. For example, although the width of the belt and the thickness of the crankshaft pulley are theoretically related, in the hybrid case it is really the clutch system that determines the pulley thickness because it is always thicker than the belt given the current technology available, so the relationship is not shown in the DSM.
4.2.2 Results and case conclusions
Here are the DSMs built through the interviews and after clustering.

![Initial DSM for the conventional thermal FEAD System (40x40)](image)

Figure 4.2.2.1: Initial DSM for the conventional thermal FEAD System (40x40)
Figure 4.2.2.2: Manually clustered DSM for the conventional thermal FEAD System (40x40)
Some interesting comments and conclusions can be drawn from the analysis of the conventional thermal system.

- The FEAD system is an inherently coupled system: the interrelationship pattern between design parameters is so that the parameter based DSM cannot be decomposed into independent cycles. That is why the final DSM presented above is not fully block triangular, and why it had to be manually clustered instead of automatically processed by the DSM partitioning algorithm.

- However, the density of the interrelationship pattern for the FEAD, measured by the Whitney ratio (see Chapter 3 section 3.4) is significantly lower than what has been the observed average on previous DSM projects. The FEAD system averages with 4.1 marks per row, i.e. there are 4.1 connections per design parameter, whereas the average observed on many DSM projects is 6.2, with little deviation for DSMs ranging from 20 to 1,000 elements.

- The reason for this apparent contradiction is the very particular topology of the influence network between the design parameters: as the DSMs show, design parameters are poorly linked together directly, but what holds together the system and makes it indivisible is the system-level requirements and their fulfillment (appearing in the matrix as feedbacks to check whether the requirement is met or not), e.g. the reliability requirement (of X years or Y thousand miles). This pattern appears most clearly in the geometry cluster of figure 4.2.2.2, where very few design parameters relate to others, whereas they almost all get input from the space & safety distance requirement and send feedback to it for its fulfillment. Requirements truly play the role of hubs in the network of influences between design parameters.

- Requirements get inputs from the first 15 rows of the clustered DSM, which are parameters outside the system (which influence it but do not get feedback from it, that is why they appear first in the clustered DSM), and therefore have an influence on how the system is designed, and in the first place at which level its requirements are set. Requirements can also get some input from parameters inside the system if the requirement is set once some design choices have been made. In that sense, the requirement is set given those particular design choices, and it would probably be set differently with different preceding design choices.

- Despite the intrinsically coupled pattern, two major clusters clearly appear, and the design parameter description shows that they correspond to two thematic categories of parameters: those relating to geometry and those relating to dynamics.

- If the design process is to be split to be more efficient than one big team handling the whole coupled system, this decomposition suggests that it should follow these categories. The blocks would represent two teams in charge of the geometric parameters on the one hand, and of the mechanical parameters on the other hand. However, there would still be a need for systems engineering to coordinate both teams and make sure the system-level requirements are met, which appears in the feedbacks between both blocks, and between blocks and requirements. The DSM suggests that the geometry team could somehow work upstream of the process compared to the dynamics team, which obviously needs input from the geometrical parameters to freeze its parameters. Again, this statement needs to be qualified because of the feedback from dynamics to geometry.

- The dynamics block is composed of two distinct cycles. One deals with tension and vibration in the belt, the other with the torque vibration damper parameters.
This page intentionally left blank
Analogous results follow for the hybrid system.

| Parameter name                                      | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 |
|----------------------------------------------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|

Blue cells: design parameters and quantities out of the system
Italic text: new parameters compared to the conventional FEAD

**Figure 4.2.2.3**: Initial DSM for the Mild Hybrid FEAD System (43x43)
Chapter 4 – Applications: PSA Peugeot-Citroën Case Study

Figure 4.2.2.4: Manually clustered DSM for the Mild Hybrid FEAD System (43x43)
Conclusions from the study of the hybrid system follow:

- While the dynamics cluster remains unchanged, a new cycle appears in the geometric cluster, which couples the crankshaft pulley to the clutch system than enables the engine to be disconnected from the FEAD.
- In spite of the major changes in the system induced by the new functions and constraints, its structure and the pattern of interactions between its parameters does not change much, neither in quality nor intensity. Figures 4.2.2.3 and 4.2.2.4 show that the same structure emerges from the analysis of the Mild Hybrid system, with insignificant change in average number of connections per element.
- This should convey the idea that the process for designing a Mild Hybrid FEAD should be roughly the same as the one for a regular thermal system (if the process is organized around two poles, geometry and dynamics). Therefore, the conclusion on this particular FEAD system is that incremental innovation (a reversible electrical machine in this case) has kept the current system structure and design process valid. It is a very lucky situation for the car manufacturers who can as a consequence adopt the incremental innovation with little capital investment or cultural change required.

Although the FEAD design process is not thought in terms of a sequence of design parameters to set, the above DSMs are very useful to understand the complexity and the interrelationships of FEADs. They give a clear map to inexperienced engineers or experts in other fields of what a FEAD truly is. Additionally, as explained in chapter 3, the above DSMs can help forecast the cascade of consequences (redesign) induced on the system by a modification of one particular design parameter. By following the sequence of other design parameters, and more importantly system-level requirements, that a change can affect, engineers and managers can decide to make the change or not.

### 4.3 Knowledge Management practice at PSA

The case study was also the occasion to observe the way Knowledge Management is implemented in a large group like PSA Peugeot-Citroën. Although it was not the primary goal of the case study, the author learnt both from the approach of a PhD student in Knowledge Management at PSA and from direct observations of knowledge management projects within the FEAD teams.

#### 4.3.1 Outline of the work of Barthélémy Longueville

Barthélémy Longueville is an alumnus of the Ecole Nationale Supérieure of Cachan, he holds the French Agrégation of Génie Mécanique (Industrial Mechanics) and a Diplôme d'Etudes Approfondies (DEA ~ Master of Science) in Knowledge Management. He is preparing his PhD doctoral thesis, in collaboration with the Laboratoire de Génie Industriel (Industrial Engineering Laboratory) of the Ecole Centrale Paris, in the Direction of Research and Automotive Innovation of PSA Peugeot Citroën.

He works specifically on the development and usage of Knowledge Management models and techniques in the design of radically innovative products. His presence in the same unit as the one the DSM project was planned to take place at, was a major incentive to conduct the January case study at PSA Peugeot-Citroën.
Unfortunately, since the DSM method cannot be applied as such to totally innovative products (see Chapter 3), there was not much room for interaction during the conduct of the FEAD structural analysis. Nevertheless, ideas were exchanged on respective approaches to Knowledge Management.

His main finding on the management of knowledge for radically innovative products follow. The conclusion of his work (back in January of 2002) is that what should really be tracked and recorded for completely new products or systems is the decision flows and their rationale. Information flows in the process cannot be studied effectively because the process itself is not well shaped yet. Conversely, decisions are useful to record.

- The design is subject to so many variations and uncertainties that sometimes teams forget why they made such and such decision, and might take further decisions with contradictory assumptions to the first ones.
- In the type of complex systems that cars incorporate, with many parts or segments developed in parallel, it is crucial to make sure that the rationale of each team is consistent with that of the others.
- It is also necessary to record the rationale for each decision in order to be able to examine the process backwards if some major issue comes up in the end. This way the source of the problem can be identified more easily, and hopefully separated from the other sets of decisions. Learning by experience is therefore facilitated.
- Finally, recording the rationale of decisions may help save time in equivalent processes for other products. Understanding what were the arguments used in the design of earlier products, as well as the consequences of each decision made, should facilitate the process for later innovative systems.

### 4.3.2 Observations on Knowledge Management at PSA

Before entering this section, it should be made very clear that a case study on a particular system for one month only is not enough to assess the Knowledge Management strategy of a large group like PSA Peugeot-Citroën. The following points should hence be taken as scattered observations the author made through the internship, rather than a coherent overall assessment that would reflect the reality of Knowledge Management at PSA.

That being said, it nevertheless clearly appeared that knowledge and its management were a deliberate area of focus of PSA's employees. First of all, as far as research is concerned, the fact that the Direction of Research and Automotive Innovation holds and collaborates to the PhD thesis on “Knowledge Management in the Design of Innovative Products” of Mr. Barthélémy Longueville, shows the interest of the group in the field. Additionally, Catherine Boulanger accepted a DSM project very quickly, even though she and Barthélémy Longueville did not know anything about the DSM method in the first place.

As for knowledge management in the daily work of the company, two rationales were presented to the author to explain why the group had begun to implement company-wide Knowledge Management techniques.
Applications: PSA Peugeot-Citroën Case Study

- **Internal information sharing for better coordination and coherence**
  Obviously, designing such a complex product as an automobile, with highly complex subsystems, requires a lot of coordination and information-sharing. A good way to help this process happen is to document a lot of the knowledge and know-how and make the documents accessible to the teams.
  It was obvious that designers were strongly committed in that direction and spent a very large amount of time to compile documents and have team debates to validate them or not. It was clear that they do not see this task as an additional burden on them but rather as a truly entire part of their job. Ensuring continuity of the knowledge by recording one’s expertise for future reference, either for oneself or for someone else in the company, seems to be regarded as a gratifying recognition of one’s expertise. It also has to do with participating to the accumulation of knowledge in the company, that is ensure a sustainable growth. Another comment heard from designers and experts about the documenting process is that they do learn about their own job and expertise when they actually compile those documents and discuss their content in team meetings.

Various types of documents were filed: (non complete list)
  - Meeting reports
  - Functional Analyses (*Analyses Fonctionnelles*), which detail the different core functions that a system has to produce and deduces general architecture and technical solutions from their analysis
  - Specifications documents (*Spécification Technique Détailée*)
  - State-of-the-art reference documents (*Référentiels de Conception*), which clarify common rules and best practices of design for a particular system, and record return on experience conclusions
  - Task technical flow charts (*Organigrammes Techniques des Tâches, OTTs*), which show in the form of a flow chart the series of tasks that have to be achieved at different levels (system level, subsystem level, part level) to design the system. See a detailed analysis of this type of document further below.

Finally, some very powerful tools were used to make all these documents available to whoever needed them. In particular, the Mild Hybrid Vehicles project team had set up a very useful intranet website (with different levels of authorized access to certain confidential documents), where absolutely all the documents were saved, referenced, and accessible through a multiple choice search engine. This way, team members working in different buildings or sites all had instant access to any document they needed for reference, from 1998 Functional Analyses to last week’s team meeting report.

- **External information flows with suppliers**
  Another important justification for documentation has to do with relationships with suppliers. When the design or manufacturing of certain parts is outsourced, very detailed information, mainly embedded in specifications documents, need to be transmitted to the suppliers so that they know what they are expected to do and under which set of constraints.

Such external aspect of Knowledge Management takes an even more critical importance at PSA Peugeot-Citroën since the group-wide policy is now to outsource entire functions rather...
than just parts. That is to say that suppliers do not have the responsibility to make proposals just on ways to design or manufacture parts of a system, but also on the more general architecture of a system entitled to deliver a particular function. For example in the case of the accessory drive system, the current outsourcing policy is to submit to the belt supplier all the characteristics of the system that PSA designers have decided of, e.g. sizes of pulleys, location of the idler and tensioner pulleys. The responsibility of the supplier is to come up with a proposal of design for the belt (material, length, width, structure, etc.) that will meet PSA’s requirements. In the next step of function outsourcing presented above, the whole function of driving the accessories should be outsourced. The supplier will then receive constraints of location of the engine and other accessories that are out of the system, but it will be its own responsibility to decide of where the idler or tensioner should be located, of the size of the pulleys, in addition to its traditional expertise work on the belt itself.

Such a revolution of the outsourcing policy induces deep issues of knowledge management.

- How is the transition managed? Transition is delicate because at this point PSA has more knowledge than its suppliers on the way to design its systems to deliver required functions, but it will gradually be the suppliers’ responsibility to design them. As a consequence, the new relationships with suppliers will lack trust for a while and their work will be carefully verified: work will have been done twice or in partnership, and both are more time consuming than before at PSA.
- How will PSA control the work of suppliers after some years of knowledge transfer? Will PSA have kept enough knowledge and know-how internally to evaluate the designs proposed by suppliers, or will they have to rely more or less blindly on them?
- In the meantime, will suppliers use the knowledge they get from PSA to sell their services to competitors? In other words, will PSA’s knowledge diffuse to competitors?

The way PSA has decided to cope with some of those issues is to have a group-wide push for documentation. In addition to the internal reasons for knowledge recording, it is crucial for PSA to record as much knowledge as possible to be able to control the work of the supplier during the transition and later on. In that context, state of the art assessments are key documents. The group-wide decision to build for every system in the company Task Technical Flow Charts (Organigrammes Techniques des Tâches, further referred to as OTTs), providing answers to general process questions (what?, how?, why?), coincided with this major shift in the outsourcing policy. From what could be understood of the strategy, OTTs are viewed as the support to transfer knowledge to the suppliers, control their work and evaluate their process, potentially help them identify weaknesses or reasons for failure. Explicitly, the author heard in the observed meeting that the purpose of OTTs was to record enough knowledge to make it possible for non-specialists to go through the design process. This is the usual illusion of process mapping: a well documented process does not mean that no knowledge is necessary to go through it.] Finally, the OTT documentation process was undoubtedly an excellent way to have teams reflect on their process, record their expertise and even gain knowledge through team interaction.

However, the limited amount of time spent on OTTs still gave the author a chance to compare the framework of the OTTs to the DSM method that was simultaneously used to study the FEAD. Again, it is important at this point to recall that the present remarks,
especially the following ones, are not based on a complete and profound study. In the perspective of the philosophy of CIPD’s Information Flow Modeling initiative (IFM), which considers information flows to be the key concern for managing efficiently product development processes, the way OTTs are currently constructed includes major flaws as compared to DSMs. Figure 4.3.2 presents the general structure of OTTs.

![Diagram showing the general structure of OTTs]

**Figure 4.3.2 : General Structure of OTTs**

Some additional constraints are put to make the chart readable: the number of cross-level feedbacks represented is limited to three, and intralevel feedbacks are simply not represented, otherwise OTTs would look like a web rather than a flow chart. For example, if task n provides an output data 2 that is not acceptable, some rework will be necessary and some upstream task, say task 1, will have to be performed again and the process will restart. However, the OTT of the system will not include such a possible feedback from the output of task n to the input of task 1.

Such representation choices induce major flaws in system analysis on the basis of OTTs.

- OTTs fail to represent the structure of the design process, because the complex interactions between tasks are not included. As explained in the description of the CIPD IFM philosophy in chapter 3, interactions are key elements of the design process, those that make it tedious and complex, and that provoke unexpected costs and delays.

- As a consequence, OTTs fail to provide a basis for system analysis. In fact, OTTs only represent the process when everything works well, with no unexpected result, rework or design loop, which literally never happens. OTTs attempt to represent the design process as a pure waterfall process by selecting the interactions it represents, but, unfortunately, there is no such thing as waterfall product design, especially for such a complex product as a car. In particular, OTTs fail to serve as project management tools to help forecast the
duration and cost of the process, because crucial feedbacks that drive those parameters are simply not represented.

- Additionally, OTTs are not that easy to read, which was actually the reason mentioned for not representing feedbacks. The waterfall structure makes the final document for a system like the FEAD so big that it is hard to capture the overall structure of the process. The FEAD system OTT took 14 letter-size pages...

A lot of these concerns were expressed by the team members themselves while reviewing the chart. Some questions like “what’s the point to represent some links and not others?” or “isn’t there a feedback test at this point?” were answered by “yes, of course, but we are limited to three cross-level feedbacks, and we have already used them...”. Some frustration could be felt in the meeting, especially when the person who led the meeting explained that it was even more difficult because the group-wide imposed formalism had changed already several times.

The process itself of building those OTTs merits a few specific comments.

- It would have been much simpler if the teams had been trained to the formalism adopted by the methodology department, they would have understood more the purpose of the exercise and its benefits. Some questions may also have been raised during the training sessions, potentially identifying the major flaws explained above.

- It would actually have been more efficient to have each OTT team meeting coached by an “OTT specialist”, who would have guided the experts through the exercise and helped them figure out which feedback they wanted to display and which not.

- Also, such an external point of view would have helped avoid approaches heard, on the tone of jokes but who knows?, like “we should not get in too much detail in that area of the process, because sometimes we don’t have time to do everything, so if we put it all in the OTT we might be blamed for not doing everything” or conversely “this task should be included because it justifies investments that we might not be able to justify to management otherwise”.

To conclude on OTTs, the author found that they were emphasizing a very unfortunate situation, in which the group clearly identified the critical issue of knowledge management associated with the decision to outsource functions, in which designers and executives are motivated by Knowledge Management and ready to spend a large amount of time on it, but in which the choice of the framework critically undermined the utility of the approach.

### 4.4 Chapter Summary

The case study conducted in January of 2002 at PSA Peugeot-Citroën is a good illustration of an interesting structural analysis using the Design Structure Matrix method presented in Chapter 3. The internal structure of FEAD systems of conventional thermal engine vehicles and Mild Hybrid vehicles was compared. It appeared that although the technological innovation from thermal to Mild Hybrid had changed the system significantly, its structure had remained very similar, enabling the current design process to be pursued. This case shows how helpful DSMs are to study the structure of any kind of system. This case was also a good example of how issues of Knowledge Management presented in chapter 2 articulate
in the real world. The author found a situation in which, in his mind, a misleading framework was used to record process knowledge and proposed the DSM framework to do so. Chapter 5 shows through a larger case study at Ford Motor Company on a more complex system how Process Mapping using DSMs simultaneously triggers efficient Knowledge Management, together with improved Project and dynamic Organizations Management.
Part III: Practical Work
5 DESIGN PROCESS ANALYSIS AND IMPROVEMENT AT FORD MOTOR COMPANY

This chapter presents the application of the approach and methods described in the previous chapters to analyze and improve the end-to-end design process of a major car subsystem at Ford Motor Company. The project was conducted over the summer of year 2002 in Dearborn, MI, while analysis, improvements and follow-up meetings kept occurring during the academic year of 2002-2003. The results and accomplishments of this project constitute the main evidence to support the point of this thesis, which is again to demonstrate how process mapping, organizations management, project management and knowledge management can be performed efficiently all at once.

5.1 Background

The choice of the study described below resulted from a very lucky coincidence between the timing of the internship and that of a process analysis and improvement task force of which the author’s research advisor had become aware. It was selected for its product development process analysis potentialities within the research interests of the Center for Innovation for Product Development at MIT. The particular commodity that was designed through the process, and even the company, Ford Motor Company, were not major factors in the study selection. In other words the purpose of the study is neither to compare the design process of the commodity with that of others, nor to emphasize Ford’s design process advantages and drawbacks as opposed to other car manufacturers’ or other industries’. Rather, the intent is to draw from a real process a set of learnings on generic issues in large-scale product development processes, and to propose a methodology to address them.

Also, in this chapter, the word design can refer to two different things depending on the context. It refers on the one hand to a general understanding of “design”, i.e. product development, used repeatedly in this thesis already. On the other hand, “design” can refer to the design engineers, the design teams, in charge of designing parts or systems as opposed to manufacturing engineers or CAE engineers, who are all part of the general design process in the sense above, but do not technically design parts or systems. The author has tried to make the context clear enough every time to avoid ambiguity.
5.1.1 Ford Product Development System (FPDS)

Before diving into the case that supported the study, it is useful to describe the general product development process followed at Ford Motor Company. Ford launched its “Ford 2000” reorganization plan in January 1995 to cut costs and boost profits. It comprised among other things promotion of computation tools and techniques – computer-aided engineering (CAE), design (CAD) and manufacturing (CAM) –, consolidation of Ford North-American and European operations to eliminate duplication in product development, and major changes in organizational and staffing policies. Embedded in the plan was the reengineering of the product development process into a new well defined structure common to the whole company called Ford Product Development System (FPDS). The intent was no different from product development process reengineering in other companies: reduce cost, reduce time-to-market, reduce staffing and increase quality. For more details about the Ford 2000 reorganization plan and the launch of FPDS, see Lewis, Winner and Reed (Lewis, Winner and Reed, 1999).

The following description of FPDS is based on documents reviewed at Ford Motor Company as well as excellent summaries by Dr. Qi Dong (Dong, 1998, pp. 26-36) and Dr. Darian Unger (Unger, 2003, pp. 128-131), both former students of the Center for Innovation in Product Development at MIT.

FPDS is a well-documented vehicle design process that includes major actions linked by reviews and iterations. FPDS follows a V shape, the architecture (vehicle level design) being cascaded to the system, subsystem and component level detailed design before later integration.

![Figure 5.1.1: The Ford Product Development System](source: Dr. Darian Unger)

Timing indicated for a S6/P6 program (see below)

The process begins by defining requirements for a new product, incorporating customer needs and engineering specifications. Specifications are defined and frozen in order, vehicle design specifications defined first and component design specifications defined later. Milestones, shown at the top of figure 5.1.1, mark progress. Each milestone includes a
review and has defined deliverables. For example, to pass the product readiness milestone, a program must have a full vehicle analytical sign-off, a confirmed and issued launch plan, CAD files reflecting verification changes, and several other key deliverables. Reviews alternate between “hard” reviews that serve as engineering tests and softer ones that serve as assessments of progress. To prepare for milestones, project team members self evaluate their performance and progress on the required deliverables by milestone time using colors (green for fully satisfied, yellow for partially satisfied, red for not satisfied). The project effectively passes the review if all required items are green.

The duration of the process varies depending on the depth of the project. The process is scaleable from the most complex level of an entirely new platform (S6) and powertrain or engine (P6) (41 months of development) to the simplest level, typically a carryover vehicle (S1) with only peripheral new parts and designs, and no engine or powertrain change (P1) (18 months of development).

5.1.2 Commodity / product studied: scale
Because of confidentiality agreements with Ford Motor Company, the particular subsystem of the car whose design process was studied cannot be revealed. However, in order to help the reader understand the scale of the project, the commodity studied can be compared to several vehicle subsystems with similar characteristics and complexity: the car body, the instrument panel with all related accessories (heating controls, radio, onboard computer, etc.), the engine, the transmission, or luxury seats (with heating, several electrical motor driven settings, multiple users memory features, etc.). These systems, including the particular one that served as the case for this chapter, all share a high level of systemic complexity. This typically means that their components are closely interconnected and interface in a way that makes the overall behavior of the system difficult to control and predict, and therefore problematic to design. They are composed of several hundreds of parts, and their supply chain therefore involves many suppliers, which means a tremendous amount of coordination required to design them. As a consequence, it usually takes several years to design these systems from scratch, and the process involves several hundreds of engineers. Finally, such systems are so important that they are made in dedicated factories or large shops with large fabrication and tooling capital expenses.

To summarize, the studied design process of the vehicle subsystem, further referred to as “commodity”, very well fits into the definition of large-scale design processes given in the introduction chapter.

The commodity is important enough that its division within Ford has developed an additional layer to FPDS which follows a set of milestones and reviews performed internally to the division and which are designed to support the vehicle milestones.
5.1.3 The need for a process analysis and improvement project: Commodity program and task force achievement history

The specific program (i.e. project to design one model of the commodity) within which the study took place, further referred to as “Prog”, was a critical one for Ford. Its goal was to design a brand new model from scratch, which happens infrequently for each category of architecture of this commodity, as opposed to more frequent programs that improve an existing model by making minor redesign efforts. In the FPDS vocabulary, Prog was on the level 6 of program complexity (S6 / P6). The outcome of Prog was therefore to drive Ford’s portfolio of the commodity for some considerable time into the future. Considerable resources had been committed to Prog, some of the commodity division’s best and most experienced program manager, program supervisors and engineers had been assigned to it and expectations were really high.

The experience of the program leadership and their interest for management issues made them carefully think about the process they were to follow, and some significant improvements from Ford’s regular way of designing the commodity were already implemented right from the start of Prog. Purchasing, Manufacturing, Quality and Testing engineers as well as a Finance team dedicated to Prog were for example all co-located with the design teams so as to enable cross-functional interaction and promote problem-solving rather than more isolated functional thinking. Some organizations however remained outside of this co-location move, like CAE engineers, who were nevertheless located in the same building close by. It is worth noting that unlike most other functions listed above, CAE engineers were not dedicated to Prog full time. Prog had one fully dedicated CAE coordinator, whose job was to assign resources from the CAE department as needed.

Weekly meetings of many Program Activity Teams (PAT) were also scheduled. PATs are cross-functional teams that focus on a program aspect requiring collective management (planning, targets, prototyping, etc.), or on a systemic outcome that is visible to the customer and that counts in his perception of product quality (e.g. NVH – Noise, Vibration and Harshness) or else on a geographic zone of the commodity that requires intensive coordination between the design teams of various parts (issues of alignment, watertightness, etc.). These organizational enablers were already an important shift from Ford’s traditions, a move from a more functional toward a more matrix type of organizational structure.

However, important process factors remained beyond the control of the leadership of one program among many others. Analysis based on a few engineers’ experience was insufficient to yield further process changes right from the start. Prog’s leadership therefore put together a task force to analyze the process, improve it internally and provide substantial data to support more important changes beyond the scope of a program and thereby seek approval from higher-level management. The task force was championed by the program manager and led by one of the program supervisors, the most experienced one who happened to also have had experience in process thinking over his career at Ford. The team comprised all functional leaders involved in the program (see list below). Some members were also from outside Prog like the commodity division quality office representatives who could provide methods and who had an interest in following the work of the task force in order to make it a best practice throughout the division if it was to prove successful.
Design process analysis and improvement at Ford Motor Company

Here is the exact composition of the task force.

**Team membership:**

Members belonging to Prog:

Full time:
- Prog program manager
- Prog program supervisors (system engineers)
- Prog Quality engineer
- Prog Timing, Building and Logistics engineer
- Prog Manufacturing engineers
- Prog Development engineers

Part time assignment to Prog (working on other programs simultaneously)
- CAE Engineers
- Planning engineer
- Testing engineers

Members from outside Prog
- Commodity Division Quality Office Representative
- CAE Division manager
- Process Excellence / KBE (Knowledge Based Engineering) support team members
- Antoine Guivarch, MIT Intern

The official charter of the task force was put the following way.

"**Problem:** [Commodity] Product Development Process is not as effective as it could be and some elements are not working.

**Objectives:**
- Align / integrate targets, assumptions, quality and other [commodity] processes with build based timing
- Reinforce targets, assumptions, design, development / design verification processes
- Alignment / commonality across the various [commodity] systems [developed in the company] with process

**Team Mission:**

[Team] Customers [are]: [Commodity division]

Product [delivered]: Clear, consistent, timed [commodity] PD process that is metric defined and aligns with FPDS [latest version specific to the commodity] or modifies it if necessary

Standards: Fairly high level, simple and not over detailed defined process and aligned timing"

The team has been meeting weekly since January of 2002 except when occasional scheduling conflicts interfered. All this had already been going on for several months when the author joined the task force in early June 2002, with a specific role to provide the team with methodologies that would be useful to analyze and improve the commodity PD process. Also, since the author was the only one dedicated full time to the task force objectives, the intent was to make the task force project move faster toward its goals.

The process issues known and identified when the task force was created or during the 5 months that preceded the start of the case study were: (some of them included in the task force charter, some of them not)
• **Unclear process of starting a new program:** commodity programs can be initialized by several different players and many different ways in the company (vehicle programs, commodity division, Ford Research Lab / Advanced Engineering that has reached a point in technology research that triggers a new program, etc.). Not all provide the design team with clear set directions required such as very basic commodity architecture. Because of conflicts between these different players, still unresolved when the programs are kicked off, major architectural changes usually occur in the programs (e.g. sedan vs. hatchback for a car body, 4 gear vs. 5 gear automatic transmission, 4 valves per cylinder V6 vs. 3 valve per cylinder V8 engine, etc.), which literally means starting from scratch again. A lot of frustration is created within the design teams because of the radical changes higher-level management imposes on programs once they are started. The feeling is that such basic planning of program portfolio should be done much more rigorously at the beginning and never changed once a program is launched. For clarification, the issue at stake here is not a change in commodity architecture because of poor testing results or the discovery of a major design issue, but rather literally a change in the program nature imposed by the commodity division or the vehicle level. This early process issue had affected Prog particularly, with several months if not a whole year wasted in a sequel of changes in the most basic architectural directions. The perception of the author might be biased by the great deal of frustration that Prog staff had accumulated over the issue, but the feeling was that this lack of strategical planning of program portfolios was something recurrent and common to all large S6 / P6 programs at least in the division.

• **Inconsistency of FPDS and its commodity-specific layer with the prototyping timing:** Reviews both at the level of the commodity and at the vehicle level do not align in a sensible timed manner with the moments when commodity prototypes and test results are available. Review outcomes being decorrelated from these major design evaluation events, their relevancy is questionable. Rework in the phase preceding the milestone might be necessary even if the review ending it has already been passed successfully, simply because prototype tests results were not available at the moment of the review. Although all necessary information was not available for the review, the milestone would still be passed only not to delay the vehicle program progress.

• **Psychological distortion on design teams resulting from the milestone system:** experienced system engineers had noticed over years that the review system introduced by FPDS had perverted the way engineers were thinking about their work. Instead of being work oriented, they gradually shifted to being review metric orientated. In other words, PD practitioners gradually evolved from a “do the work” approach to a “meet the metrics” approach. This is a significantly different working process. One experienced Prog program supervisor sketched the diagram of figure 5.1.3 several times to explain what this psychological evolution meant for the work pace. When the milestone review is perceived far away on the road, work load is under what it should be as nobody is behind the back of the designer to tell him that his component will not meet the milestone. But as the milestone gets closer, panic makes its way through people’s minds and they start working overtime to meet the milestone requirements on time. As soon as the milestone is passed, the pace shrinks again.
• **No work roadmap between each milestone:** what needs to be achieved by the next milestone is known, but the path of necessary actions to get there is not documented and only very experienced system engineers might know enough to drive the process to its goals. A MIT student once said while conducting a case study at Ford: “FPDS is not a process, it is merely a schedule.” Most engineers involved in the process do not have this vision of next steps coming and they are therefore likely to lack anticipation or be short-term focused. In other words, FPDS is a checkpoint process that focuses on the sequence of goals to be achieved but does not inform on the work to be accomplished to reach the goals. This point connects with the previous one: Prog leadership felt FPDS had put too much emphasis on reviews and metrics, and not enough on work.

• **Misaligned timing between different functions involved in the commodity PD process, inefficient coordination:** because of the lack of big picture cross-functional workplans between the milestones, several functions or subprocesses (CAE or Testing for example) have defined their own internal workplan. It is obvious to anyone who has worked a couple of months in PD at Ford that these various workplans are often misaligned, resulting in chronic delays in required information flows and therefore either delays, or low quality work done in the last minute for the program. This situation was reported several times to the author, with several functional departments as examples. The following illustration was told by a CAE engineer: component engineers keep calling CAE engineers asking for models, but CAE engineers haven’t even received the basic assumptions they need to create realistic models by the time component engineers already need them. The only conclusion that both can draw is: “we’re in big trouble”...

The author has found interesting quotes on this general issue of faulty coordination by the FPDS Communications group manager at the time of Ford 2000 reorganization launch, David Roggenkamp: "You might call one of your functional colleagues to discuss an idea or to find out if a product problem has already been solved. But by the time you've left a few voice messages or e-mails and still haven't heard back, you whip out a blank piece of paper and figure it out yourself." (Williams, 2000) The response that the Ford 2000 reorganization put forward was the creation of online communities in
various forms, which had been fully operating for several years already at the time the author conducted the case study. It looks like the identification of the issue was correct, but the diagnosis was somewhat wrong, and therefore the remedy pretty inefficient.

The extent to which the commodity PD process participants lack common understanding of their connections will be quantified in detail in the analysis section 5.3. Timing disconnects will be dealt with later in section 5.5.

Consistently with this analysis of the situation, the task force decided to adopt an approach based on the work each functional department is required to provide to others in order for the process to move forward. The vocabulary of “deliverable” was adopted to convey this idea: a deliverable is a piece of information, often formalized into a standardized document but not necessarily, that one group delivers to a set of others. A presentation of the task force work to higher-level management stated: “Workplans are the foundation of success. They are used to forecast potential issues, contrary to after-the-fact reaction based management. Workplans are driven by [prototype] builds deliverables. (...) A “Gives and Gets” philosophy is used to develop a total program team understanding of deliverables and information flows required at different points in the process to enable successful overall plan. Communication and data are keys”. The details of this approach will be described in the following section 5.2.

It is important at this point to stress again that the intent was to understand the process and align the various functional subprocesses together. The task force was well aware of the trap to go into too much detail in the description of the workplan, rigidifying the process to an extent that it would have no flexibility anymore, producing an over detailed document that nobody would use, and restricting the applicability of the changes to Prog instead of the whole commodity division. As stated earlier, this concern was embedded right from the start in the mission statement of the task force and remained in the team members’ mind throughout the project. Ford Motor Company had conducted many process improvement projects in the past, especially in the commodity division, and several of them had yielded no result at all indeed, because of the fault described above. The nightmare of a particular project called “Sequential Design Freeze”, which had produced a deck of dozens of pages of process documentation that nobody ever read, was constantly reminded to the task force throughout the weekly meetings it held.

In its initial phase, the task force reviewed subprocesses individually (usually tied to a single functional organization, but not necessarily, for example design, development, testing, durability, CAE, Target Development, Sourcing, Quality, Program Approval, Assumptions, Build, Manufacturing). The intent was to identify key subprocess internal issues (like misalignment with the prototype build schedule), but above all to have all other subprocess owners hear the presentation and identify potential interface issues with their own subprocess. This first phase was helpful to identify some additional common issues, but it quickly proved too slow to review only one subprocess a week, and too difficult to get deep enough in the cross-functional interface issues in one hour of meeting. That is when the process moved to a larger data collection enterprise described in the next sections.
5.2 "Gives and Gets" data collection framework

A one day offsite meeting with all major subprocess owners was scheduled in early June. Participants were mainly the task force team members, but relevant others were added, like Prog Purchasing engineers and others. It happened very luckily to fall on the second day of the author's stay at Ford to conduct this research.

Every participant was given two sets of blank cards to prepare for the meeting, some blue, some yellow, and was asked to fill them with one deliverable per card that, respectively, they needed from another functional team (blue card, a "get") or were providing to other functional teams (yellow card, a "give"). On each card would be detailed the team providing the information, the delivery date in months before job 1 (i.e. months before production launch), the team(s) receiving the information and a description of what information is transmitted.

A fictitious example helps the reader fully understand the terminology used throughout this section and the next. The process owner of organization A, which has an input and output configuration as shown in figure 5.2.1, brings three get cards (blue) for deliverables 1, 2 and 3, and three give cards (yellow) for deliverables 4, 5, and 6.

![Diagram](image)

**Figure 5.2.1**: Fictitious example of the input (get) and output (give) scheme declared by team A

The card for deliverable 1 is blue for a Get card, and would look like figure 5.2.2.

| Source: B |
| Delivery date: xx months before job 1 (i.e. when A needs the information to be sent by B) |
| Deliverable description: the type of information that A needs from B in deliverable 1 |
| Recipient: A |

**Figure 5.2.2**: Example of a "get" card

And similarly for deliverables 2 and 3.
On these get cards, recipients of the deliverable other than the team who requests it are not listed (or several get cards for the same deliverable were declared by each different recipient). In the example above, A does not know or care whether other teams than A also receive deliverable 1, and therefore only lists “A” as recipient.

The card for deliverable 4 is yellow for a Give card, and looks like figure 5.2.3.

Source: A  
Delivery date: yy months before job 1  
Deliverable description: the type of information that A provides to C and D in deliverable 4  
Recipients: C, D

*Figure 5.2.3: Example of a “give” card*

And similarly for deliverables 5 and 6.

On the give cards, all recipients known to the source are listed.

What makes a give different from a get is not which team is the source and which team is the recipient. It really is who writes the card. If the writer of the card is a representative of the source team of the deliverable on the card, then the card is a give. If the writer of the card represents a recipient team of the deliverable, then it is a get. For example, the card shown on figure 5.2.3 could very well be a get with the exact same content and information written on it, if either C or D teams representatives had written it.

Teams B, C and D also prepare their own cards similarly, without discussing across teams, and the purpose of the offsite meeting is to compare each team’s expectations with the perspective of the other teams. The comparison terminology is defined in figure 5.2.4.

*Figure 5.2.4: “Gives and gets” analysis terminology*
For each team, A in the example of 5.2.4, give cards declared by the team (for A: deliverables 4, 5 and 6) are compared with get cards claimed by other teams to request information from it. Figure 5.2.4 only shows this comparison with team B’s expectations, but the same comparison takes place for teams C and D, in order to know if the give card that A declared for deliverable 4 was matched with two get cards claimed by C and D for this same deliverable 4.

When the give card is not matched with a similar request by the team who is supposed to receive the information, the deliverable is called an "unmatched give" (deliverable 5 in figure 5.2.4). This simply means that the source team declared it was providing a team with some information, but that the receiving team apparently did not need such information because it did not request it from the source. Further interpretation will be given in the next section.

If a give card and a similar get card referring to the same deliverable description from the same source matched, both cards are merged into a single deliverable called a "match", like deliverable 6 of figure 5.2.4. It is not noted on the chart, but deliverable 1 is also a match because it is declared by A as a get card and by B as a give card. This situation means that there is an agreement, a mutual understanding, between A and B about the information exchange. At this point, whether the delivery dates on the cards that match are identical or not is not taken into account. In any case, it is considered a match.

Finally, if a get card is not matched by any similar give card, then the corresponding deliverable is called an "unmatched get", like deliverable 7 in figure 5.2.4. It means that the need for information of the team who declared the get card was not anticipated by the source.

Further interpretation and explanation of these phenomena will be provided in the next section, which presents the real data.

5.3 "Gives and Gets" analysis results for the commodity design process

Back to the commodity design process, it is helpful to give a few examples of what the deliverables described on the cards were.

- Target one-pager from the testing team to the assumptions team (give)
- List of recommended suppliers from component engineering to purchasing (get)
- Prototype build design freeze from systems engineering to purchasing (give)
- Bill of materials from procurement and build team to purchasing (give)
- Component FMEA (Failure Modes and Effect Analysis) from component engineering to quality (get)
- Durability analytical model from CAE to component and systems engineering (give)

When they were asked to prepare their cards, team representatives were clearly told that the base timeline for the exercise was a S6 / P6 program in the FPDS scale (see section 5.1.1 for a definition of these terms). They were also explicitly asked to fill the cards based on real work, not on FPDS events scheduled at the reviews. Two teams unfortunately simply wrote
on cards the FPDS deliverables, which was corrected later. Also, teams were not supplied an exhaustive list of the teams they could use to name the source or the recipients of the information. Some bias resulted in this lack of preparation, which will be examined further.

The list of distinct teams that were represented in the meeting and whose deliverables were collected is shown in table 5.3.1.

| Assumption setting (distinct from Sys. Eng.) | Planning team (including all higher-level teams like commodity division & vehicle) |
| Target setting (distinct from Systems Eng.) | 5 distinct teams within CAE |
| Systems Engineering | 3 teams within Testing |
| Component Engineering (no distinction between components) | Manufacturing Engineering |
| CAD design team | Prototyping Shop |
| Prototyping procurement & logistics | Purchasing |
| Quality | Supplier Technical Assistance |

**Table 5.3.1: List of teams represented at the offsite meeting**

Figure 5.3.2 shows the practical way the meeting was conducted. A large wall was prepared to receive the numerous cards written by the team representatives. On the horizontal axis time was figured in months before job 1, with the main checkpoint reviews clearly identified, and on the vertical axis the different functional teams were listed. At the beginning of the meeting, team representatives were asked to nail their cards with the following rules: cards had to be put on the row of the source of the deliverable and in the column corresponding to the delivery date. Therefore team representatives put all their give cards (yellow) on their own row, and put their get cards (blue) on the rows of the teams they were requesting information from. So that in the end, the row of each team contained all the deliverables they were supposed to deliver to others, whether the team had put the card on its own work itself (yellow give card) or some other team had requested information from them (blue get cards). On the fictitious example of section 5.2, the row for A would hold yellow give cards 4, 5 and 6 as well as blue get cards 6 and 7 put by B and any get card that C and D would have prepared to request information from A. Get cards filled by A for deliverables 1, 2 and 3 would have been nailed respectively on the rows of B, C and D. The comparison accomplished on figure 5.2.4 could therefore easily be done within each row, examining only the outputs of the corresponding team.
Before showing and interpreting the data gathered at this meeting, let us review each category of deliverables found and think about all the possible causes for their apparition. Readers who are not interested in that level of detail in the analysis might want to skip the following cases until the middle of page 67.

**Unmatched gives**

A deliverable can turn out to be an unmatched give if:

- **The information is indeed delivered to a recipient that does not really need it to accomplish his work.** It can be either a completely useless information flow, that does not help the recipient in his work in any way, or some data distributed just for the information of the recipient, but that does not constitute a real input to his work. In that last case it is still useful to the recipient to know what is going on in the program, although there was not a formal request for the deliverable.

- The source team mistook a team for another when they filled their cards. In other words they thought it was the team they listed as recipient which needed the information when it is actually another one who put a corresponding get card. This situation would be caused by a **faulty understanding of who needs the information produced.** It could also be that some team denominations on the cards were ambiguous since the list of relevant teams had not been supplied to the meeting participants in advance. The staff and team representatives tried to identify and reconcile the cards falling in this category, but some might have remained in the data presented below.

- The source team representative who wrote the give card used a different terminology from that of the recipient for the same piece of information, so the cards were not identified as matching although they should have. With the help of experienced engineers, this situation was mitigated as much as possible but the author acknowledges that a few exceptions might have remained.
• The source team put lots of give cards for different subsets of the same deliverable, for which the recipient only put one get card. Again, the author tried to combine the cards falling in this category: cards were identified as matching as soon as they had at least one subdeliverable in common, but it might still be a cause of marginal bias in the data.

• The source team went into a level of detail higher than the general expectation of key deliverables that they were asked to provide. Therefore the recipient has not even declared that it needed the information because it was a secondary input to them. It could also be that the source thought the deliverable was a major output for them (for example because it requires a lot of work) while the recipient thought it was a secondary input.

• The deliverable declared by the source team is merely an FPDS event (e.g. a meeting, an approval, a sign-off) which does not contain any work. Such events were introduced in the data by teams who simply copied FPDS documentation to fill their cards, and were gradually updated in the data later on.

• The recipient team representative was not aware that his team needed the information because he works in a different subset of the team that does not use this particular data. Subprocess owners were however chosen for their experience and the overall understanding of the work of their whole team / subprocess.

• The recipient team representative forgot the need for the deliverable when he prepared his cards.

• The recipient team representative was not present at the meeting and did not bring his cards. This situation was removed from the statistics shown later on in this section. In other words only the deliverables for which both sides of the story could be compared were counted.

Unmatched gets
Similarly, a deliverable can be an unmatched get if

• The source team had not anticipated the need of information by the recipient team. It does not necessarily mean that the source team cannot provide that information, but simply that when writing the cards, when thinking of what outputs they were producing, the source team did not think of this one. In other words, the necessity for them to produce the deliverable for the recipient team was not formalized and documented in their internal process documents, if they have any at all.

• The recipient team requested the information from the wrong source: the team that the recipient identified as the source for the deliverable they are requesting is actually not the one that produces the deliverable. The real source might have put a corresponding give card that matches. The staff and team representatives tried to reconcile these mismatches as much as possible during the meeting, but some might have remained in the data. In such a case, the mismatch denotes a lack of understanding by the recipient of who does what in the process, because the source of a piece of information is identified wrongly.

• The recipient team who wrote the get card used a different terminology from that of the source for the same piece of information, so the cards were not identified as matching although they should have. With the help of experienced engineers, this situation was
mitigated as much as possible by grouping similar cards, but the author acknowledges that a few exceptions might have remained.

- The recipient team put lots of get cards for different subsets of the same deliverable, for which the source only put one give card. Again, the author tried to combine the cards falling in this category, and cards were identified as matching as soon as they had at least one subdeliverable in common, but it might still be a reason for marginal bias.

- The recipient team went into a level of detail higher than the general expectations of key inputs that they had been asked to provide. Therefore the source has not even declared that it was issuing the information because it was a secondary output to them. It could also be that while the deliverable is a secondary output for the source, it is a key input for the recipient, of which the source is not aware.

- The source team representative who wrote the cards was not aware that his team was providing the information to others because he works in a different subset of the source team that is not involved in the corresponding work. Subprocess owners were however chosen for their experience and overall understanding of the work of their whole team.

- The source team representative forgot the deliverable when preparing cards.

- The source team representative was not present at the meeting and did not bring his cards. This situation was removed from the statistics shown later on in this section. In other words only the deliverables for which both sides of the story could be compared were counted.

All other situations yield a positive match between one or several give cards and one or several get cards.

It can be seen from the scenarios above that there can be many underlying reasons for each matching / unmatching situation, which could lead the reader to think that interpretation is risky. However, an important effort was put in reducing the biases of the data by reviewing the cards for several hours during the offsite meeting (each team representative reviewed all the cards for which his team was identified as the source, and had the chance to say that some cards belonged to another team and not to his, etc.). Table 5.3.3 shows adjustments made to the data to reduce biases.

<table>
<thead>
<tr>
<th>Total number of cards</th>
<th>495</th>
</tr>
</thead>
<tbody>
<tr>
<td>Removed because the other point of view was not represented (technical impossibility to have a match, one of the involved teams representative not being present at the meeting)</td>
<td>-65</td>
</tr>
<tr>
<td>Redundant cards within the same category (redundant gives or redundant gets, i.e. additional cards describing a deliverable already recorded on another card of the same category)</td>
<td>-30</td>
</tr>
<tr>
<td>Number of relevant cards</td>
<td>400</td>
</tr>
</tbody>
</table>

**Table 5.3.3: Data Statistics and bias reduction measures**

Given this particular scrutiny on biases, the author believes that the main interpretation that can be drawn from the data is that the amount of matches measures the mutual understanding of the overall process, the mutual awareness by each team of how it relates to
others when it comes to exchanging information. The actual numbers are subject to
certainty from the noise factors described above, but the general distribution of
unmatched gives and gets and matches allows a qualitative assessment of the system-level
awareness and thinking. Figure 5.3.4 exhibits the results.

![Pie chart image]

**Figure 5.3.4:** Gives and Gets analysis results for the overall process

It appears clearly that the awareness of the process at the process-level, i.e. across the
different teams, is very poor. Only 6% of the deliverables declared by all teams represented
two-way matching deliverables. **This result confirms to an extent that was not expected
the concerns stated about the lack of common understanding of the position of each
team with respect to others in the overall process.** Some argue that misalignments are
always the result of a give and get exercise, and that the give and get process itself drives the
result, but the misalignment in this situation is serious enough to yield an undeniable
conclusion of little process-level knowledge. There was therefore a tremendous need for a
new shared agreement between all the players on what the process really is. Whether the 6%
figure reflects accurately the degree to which this statement is true is of little importance:
even a figure of 40% would be considered unsatisfactory because it would still mean that
60% of the total number of deliverables are not mutually agreed on by source and recipient.
Also, the uncertainty tied to the biases listed in the preceding pages is balanced by the
adoption of the most tolerant accounting method, which recognized matches of cards as soon
as one element of the sometimes several deliverables described in the card was matching
between a get card and a give card. This choice definitely has an importance because from
the 465 deliverables declared at the meeting (including the deliverables for which only one
point of view was available), the analysis to be described in the following pages jumped to
596 deliverables as distinct ones declared on the same card were separated. The accounting
of the matches was therefore very liberal.

---

1 A pair of give and get cards matching are counted as only one deliverable. See figure 5.3.5 for an example of
how statistics are counted. Out of the five cards represented (deliverables 4, 5, 6 and 7 as gives and 6 as get),
one pair is matching and therefore 4 deliverables are counted, yielding the percentages indicated. In the real
situation, out of the 400 relevant cards, 24 pairs are matching so the total number of deliverables is 376.
It is also remarkable that unmatched gives and unmatched gets weigh comparably in the process. The experience described earlier that engineers of different teams were running after engineers of other teams to request information—in the case when the source team was not aware that it needed to supply the information to the team requesting—supported an expectation of a high number of unmatched gets. However, it looks like an equivalent number of deliverables are supplied when nobody requests them, which means that teams also get information that they did not expect to get. When the transmitted data is useful to the recipient team (as opposed to when the deliverable is passed on for their general information), it might lead to necessary rework if the data received is new or contradicts an assumption made by the recipient team earlier.

It is important to qualify the above results so that no major interpretation mistakes are made. The only thing that the figures above measure is the mutual awareness of the teams of what others expect from them. The reader should not interpret the figures as an indication that 46% of the work done is useless (unmatched gives), or that conversely 48% of the information is missing to complete the program (unmatched gets). In the end, the product does get designed, released, sold and it surely does not exhibit the flaws that such an interpretation would make one worry about. What these figures show is that the initial vision of each team of what they should receive is very incomplete (46% unmatched gives, engineers at the meeting were positive that almost none of these deliverables were useless, even teams that had not requested them; it is consistent with the cost and time cutting measures that had been implemented at Ford and elsewhere, which would not tolerate such a high level of waste). Similarly, their initial understanding of what they need to supply to the rest of the organization is very incomplete (48% of unmatched gets). What happens in reality is that all the deliverables do get transmitted in the end after a few phone calls to resolve unplanned needs for information. The kind of exercise conducted at this offsite meeting can only be a one-time experience since one of the outcomes of the offsite meeting is to consense with everybody on each other’s needs for information. After processing the data of the meeting, each process owner has a full vision of what is expected from his team, and if the exercise was conducted again they would hopefully all reach almost 100% of mutual awareness.

Interestingly enough, this overall distribution of deliverables by category was not uniform among the different teams. Analyzing the outputs of each team allows to detect significant differences in the understanding of the process. Again, by outputs the author means all the deliverables that a team supplies to others, whether they were declared as a give by the team itself or requested by others as a get. All are assumed to be required from the source team. Figure 5.3.5 uses the example of section 5.2 to clarify what is meant by outputs.
Figure 5.3.5: Framework for the analysis of the outputs of each team

Figure 5.3.6 gives the most extreme examples of output distribution in the real process. Contrasting with the rather balanced overall distribution between unmatched gives and unmatched gets, different subprocesses can have a very different pattern of outputs. The three subprocesses shown in figure 5.3.6 have a statistical significance: CAE, Testing and Design subprocesses account for respectively 51, 44 and 129 cards out of the 376 total. The CAE chart represents the overall statistics of the 5 CAE teams combined; similarly Testing encompasses two testing teams, and Design the systems engineering, component engineering and CAD design teams.

Figure 5.3.6: Most extreme examples of output distribution in the real process
The total set of cards that was put on the row of a team on the wall at the offsite is still assumed to constitute the total amount of work that needs to be achieved by the team, whether it put the deliverable on the wall itself (give card) or somebody else requested it (get card). Within this assumption, it can first be seen that CAE, Testing and Design all do poorly in the proportion of matches, comparably to the overall process, with some advantage however to Testing. But what is strikingly different is the relative importance of unmatched gives with respect to unmatched gets.

For CAE and Testing, the work pattern can be named as "work push", because they anticipated 75 and 72% of the total number of deliverables that they have to supply to the rest of the program (unmatched gives + matches, i.e. the proportion of give cards they declared). In fact, they anticipate so much that the recipients of the information do not think they need it and do not request it. The main cause of this enormous proportion of unmatched gives is the fact that both teams detailed each and every single deliverable they were producing with one card each. The reason for that is not anecdotic, it is simply because both teams had formalized in a precise workplan their internal process (an inflow / outflow diagram made by Mr. Tachih Chou for CAE, and the Testing "Double-V" process document). They had a well defined vision of what their work process is and what type of information they have to supply to others. It was probably too detailed for other teams to think of the corresponding requests (get cards), in particular when CAE declared separate deliverables for different components, which was not the approach adopted by other teams. Component engineering was considered as one team only indeed, and its deliverables were general enough not to detail every single component except the main ones. This explains in part that the deliverables show as unmatched gives rather than matches. The key conclusion however remains that CAE and Testing teams anticipate to a large extent the nature of their work. It was pointed out to the author that this finding is not necessarily true throughout Ford Motor Company, and relies primarily on the fact that the CAE and Testing teams of the commodity division had worked to formalize and document their process. The finding is therefore not a conclusion that CAE and Testing engineers have a process-oriented culture, but rather that the particular management of those teams in the commodity division had done a good job at promoting process documentation and that it resulted in a better anticipation and process knowledge by their engineers.

On the other hand, the Design teams are asked for a lot of information that they did not expect to have to supply in the first place. The give cards they declared only account for 29% of the total amount of work they turn out to be delivering. 71% of their deliverables are requested, reminded to them by the recipients. From these figures, it can be said that what drives the design teams is the information pull from recipients, and with such an unexpected work load, it appears that they are likely to run out of time frequently and do the work in the last minute. Design teams are really the ones who centralize the information from many other teams, CAE, Testing, Manufacturing Engineering, etc. One important design team is the system engineering team whose role is to make sure that all subsystems integrate well. The scope of the design teams work is very broad and it is understandable that it is more difficult for them to have the general vision that more specialized teams like CAE and Testing had. As a consequence, they did not have a clearly documented workplan, which explains why
their cross-functional process knowledge is so poor, in spite of the cross-component knowledge their function entails.

Other teams also have different results, but they are less easy to interpret for several reasons. Sometimes the boundaries that define some teams were ambiguous to some participants and it is difficult to know if an unmatched get is due to a lack of mutual awareness or due to a different understanding of what the team encompasses. Above all, these remaining teams do not have such an important role in the process and their data has less statistical significance. The complete data is nevertheless supplied in Appendix B.

5.4 Initial offsite accomplishments & data processing framework to follow

The offsite meeting largely described in the preceding section had two major outcomes:

- Provide large quantities of data to study the cross-functional process knowledge as completed in the preceding pages
- Above all have all the teams involved consensus on a list of key deliverables that constitute the commodity design process.

Because it proved the critical necessity to conduct a process-wide education and improvement project, the author insisted quite a lot on the first point in the last pages. However, from the task force's perspective, it was really the second point that mattered. It was indeed the first step (i.e. "know where you start from", see Chapter 2, section 2.1) toward process improvement. As explained in the end of section 5.1.3, a program-wide meeting like the 'offsite' was necessary to gather everyone's stance on the process at the same time.

The cards, processed so as to eliminate redundancies and to merge matching give and get cards into a single deliverable, would therefore be the basis for the next step of the analysis, that is the resolution of the process issues identified in section 5.1.3.

As explained in section 5.3, the data that came out of the offsite meeting was not perfectly clean and ready for full analysis. Some teams had mistaken the colors of the card, some had not identified sources or recipients correctly (because no list of appropriate teams and their denominations was provided to the subprocess owners to fill the cards), some cards were holding several deliverables that needed to be separated, some information was missing (like an unambiguous delivery date or some of the recipients data). Also, some get cards were requesting information to a group of teams rather than a single source team, which was not really expected.

After all cards were typed into a large Microsoft Excel spreadsheet by the KBE team, the author reviewed the data with the task force leader and the KBE team to develop a common framework for all deliverables. At this point, the notion of give or get was abandoned since all the cards had been seen by everyone, they now constituted a consensed list of deliverables of the commodity design process, and who had declared them in the first place had no importance anymore.
Design process analysis and improvement at Ford Motor Company

Each participating team was given a 3 letter code to represent the team. This same code would be used to identify the team whether it was a source or recipient. A very clear definition of these codes and non-ambiguous boundaries of the teams they were referring to was established. Each deliverable was assigned to a unique source organization who owns it and has the responsibility for it. To take the multiple source teams issue into account, another attribute of deliverables was created: the notion of contributors, who participate in the preparation of the deliverable but do not lead the effort. Each deliverable was also given a unique identification code, called “deliverable ID”. The standard format of a deliverable therefore comprised:

- Its unique ID, in the format ‘XYZλα’ where
  - ‘XYZ’ is the 3 letter code of the unique deliverable owner
  - ‘λα’ is the delivery date (in months before job 1) attached to the deliverable (1 or 2 digits number, sometimes with a half month precision, and which can be negative)
  - ‘α’ is a meaningless letter that serves to differentiate between deliverables owned by XYZ and all delivered at λα months before job 1
- A list of contributors, possibly empty, named by their three letter code
- A list of recipients, never empty, named by their three letter code
- A short verbal description of the information contained in the deliverable, in a clear enough way for every commodity process participant to understand immediately what it refers to

The author met individually with all subprocess owners who had written cards for the offsite meeting, to review their deliverables and complete the data as necessary to fit the above standard format. In the meantime, some data was already corrected to address immediately when possible some timing conflicts identified with other teams during the offsite meeting. Table 5.4.1 gives several examples of deliverables with the signification of the team codes.

<table>
<thead>
<tr>
<th>Deliverable ID</th>
<th>Contributors</th>
<th>Recipients</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASMαααg</td>
<td>DEP</td>
<td>DCD</td>
<td>Preliminary Design Direction – Packaging Envelope</td>
</tr>
<tr>
<td>QUΑββα</td>
<td>DEP</td>
<td>DEP</td>
<td>System-level quality history review and prioritization</td>
</tr>
<tr>
<td>TRΓγγα</td>
<td>DEP, DCP</td>
<td>[Testing], DCP, DEP, PLN, ASM</td>
<td>Targets updates cascaded to support Program Approval Milestone</td>
</tr>
<tr>
<td>PBSδδδα</td>
<td>PPM, DEP</td>
<td>[Testing]</td>
<td>First prototype build available</td>
</tr>
</tbody>
</table>

ASM: Assumptions   DEP: Systems Engineering   DCD: CAD design   QUA: Quality
TRG: Targets      DCP: Component Engineering   PLN: Planning   PBS: Prototype build shop
PPM: Procurement & Build, Logistics

**Table 5.4.1: Examples of deliverables in the standard format**

Delivery dates and more technical deliverables cannot be shown for confidentiality reasons, but they are there too

In the end of this data review phase, the process analysis characteristics and outcomes were as shown in table 5.4.2.
### Table 5.4.2: Process analysis characteristics and outcomes at this point

From this set of data, more could be known on how each team works. Table 5.4.3 shows a sample of statistics that can be drawn. The complete data is supplied in Appendix C.

<table>
<thead>
<tr>
<th># of Deliverables Workstreams</th>
<th>Owned</th>
<th>% of total process</th>
<th>Contrib.</th>
<th>% of total</th>
<th>Recvd.</th>
<th>% of total</th>
<th>Total row</th>
<th>% of total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component Eng.</td>
<td>69</td>
<td>12%</td>
<td>88</td>
<td>15%</td>
<td>152</td>
<td>26%</td>
<td>267</td>
<td>45%</td>
</tr>
<tr>
<td>Testing</td>
<td>82</td>
<td>14%</td>
<td>13</td>
<td>2%</td>
<td>110</td>
<td>18%</td>
<td>202</td>
<td>34%</td>
</tr>
<tr>
<td>Systems Eng.</td>
<td>81</td>
<td>14%</td>
<td>76</td>
<td>13%</td>
<td>263</td>
<td>44%</td>
<td>378</td>
<td>63%</td>
</tr>
<tr>
<td>Manufacturing Eng.</td>
<td>51</td>
<td>9%</td>
<td>8</td>
<td>1%</td>
<td>73</td>
<td>12%</td>
<td>116</td>
<td>19%</td>
</tr>
<tr>
<td>Planning</td>
<td>62</td>
<td>10%</td>
<td>3</td>
<td>1%</td>
<td>71</td>
<td>12%</td>
<td>129</td>
<td>22%</td>
</tr>
</tbody>
</table>

### Table 5.4.3: Statistics on the contribution of each team in the overall process (out of a total of 596 deliverables)

The importance of each team in the overall process can be measured several ways. The number under the “owned” column is the number of deliverables that the team is the lead source for. In the sample data of table 5.4.2, it looks clear that Systems Engineering weighs much more than Manufacturing Engineering in terms of ownership of deliverables.

Another way to evaluate the involvement of a team is to add the number of deliverables for which they are the lead and those to which they contribute. The resulting figure measures the general contribution of the team to the process. Component engineering and System Engineering are confirmed in their central role with a 27% contribution to the overall process each, while Manufacturing Engineering and Planning, although certainly key in the process, weigh only respectively 10 and 11% in that sense. There are of course many teams that weigh much less than that, but their relevance in table 5.4.2 was questionable since their data sample was so small. Such raw statistical analysis does not take into account any difference in the importance of the deliverables, it just counts them. The reader should also be careful when manipulating the percentages above: while the percentages for issuances add up to 100% because there is always only one lead team per deliverable, the “contributed to” and “received” percentages do not add up to 100% because each deliverable can have several or no contributors and several recipients. The total number of contributions amount to 60% of the number of deliverables, yielding an average 0.6 contributor per deliverable not counting the lead team, or equivalently 1.6 participants per deliverable.

Comparing the number of deliverables owned to the number of deliverables contributed to is also very instructive on the way each team works. For systems engineering and component engineering, these figures are roughly comparable, which means that the teams are permanently involved in collaboration with other teams, as much as leading on their own deliverables, for which they might as well be coordinating with others. On the contrary, teams like Manufacturing Engineering and Planning have little involvement to support other teams because of their specialized function. It can be surprising that although Manufacturing
Engineering collaborates closely with the design teams in regular meetings (concurrent engineering), they were not listed as contributors to many design deliverables.

Deliverables received tell us how much each team gets to know what goes on in the process, even if they are not involved in the creation of the deliverables they receive. Again, the central role of Systems Engineering and Component Engineering is confirmed since they receive respectively 44% and 26% of all the deliverables produced in the process, while more peripheral teams like Manufacturing Engineering and Planning only see 12% of it. The total number of deliverables received amounts to 248% of the number of deliverables, yielding an average 2.5 recipients per deliverable (deliverables received figures is generally bigger than other figures because a lot of documents, like assumptions, are distributed to every team regularly).

Finally, the total number of deliverables that a team touches, whether owning, contributing or receiving is also shown. It quantifies the overall positioning of each team in the process. The figures in this last column are not just the sum of the other columns since there is sometimes some redundancy, contributors of some deliverables being also listed as recipients.

Overall, such statistical data proves useful to analyze the role of each team in the process. It seems that Systems Engineering and Component Engineering are major hubs in the network of teams, whatever the metric chosen to study it. This idea will be examined in greater depth in the organizations management findings section 5.6.

However, all the work done so far is of little help to understand the process itself. Plenty of data and conclusions on the roles of subprocesses or teams were drawn, but the only process outcome is a list of deliverables with a time label. The above analysis answers the who, what, and when questions, but a crucial range of data is missing: the key what for? question. In order to fully understand the process, one needs to understand how deliverables relate to each other over time, which serving as input for which. Only then can the process be sufficiently understood, analyzed and improved. That is what DSMs are used for in the next section.

5.5 Deliverable DSM analysis: tracking the information flow over time

5.5.1 Rationale
The list of deliverables gathered so far provides an overall understanding of what needs to be done to design a model of commodity, with a rough timing sequence. But to know if the process makes any sense, to identify it strengths and weaknesses, one needs to study the relationship between the deliverables. Namely, to make sure that the dates attached to the deliverables are realistic and that the process can be completed on time, the inputs of each deliverable need to be identified, and their delivery dates need to be compared with how much time is required by the team getting the input to process the data and produce their output. This is another large amount of data to collect, and a deliverable Design Structure Matrix appeared to be very appropriate to map the deliverable-to-deliverable relationships.

The necessity became clear to ask each process owner to identify the required inputs for each of their outputs (i.e. the deliverables for which they lead the work), and the lead time to process the input data before the output can be produced (figure 5.5.1 shows on an example
the data to be collected). As an implementation plan for this second phase of data collection was proposed to the task force, the program manager asked to add another interesting question to the input-output timing relationship. He wanted the recipients of an input, that is the customers of the team who produced the input, to evaluate the quality of the information they get in the input. This was in an attempt to understand which specific deliverables were problematic to the teams who use them, and to try to improve these deliverables one way or another. Adopting a customer satisfaction focus, within the process like for the end product, the approach was to ask the users of the information and not the team who delivers it about the quality of the deliverable. Cultural gaps between functions and teams (different language, different methods, different goals) and the lack of cross-functional team communication could very well indeed create a situation in which the team who owns a deliverable thinks it does a perfect job in supplying the right information to the teams who need it, while these recipients perceive the incoming information as incomplete, subject to change or irrelevant to their needs. The notion of quality is quite vaguely defined at this point, the exact wording of the questions presented in the Data collection process section 5.5.2 will help the reader understand its meaning. Deliverable owners were asked to evaluate the quality of their inputs on a Low (L), Medium (M), High (H) scale.

On top of the evaluation of the perceived quality of the inputs, one also needed to qualify the importance of the inputs’ quality. It could for example very well be that its recipient evaluates the quality of an input as low but that this low quality does not affect the future quality, in his eyes, of the output it will produce partially based on the input. Each process owner was therefore also asked to evaluate the impact of the quality of their inputs on the quality of their outputs, on a Low (L), Medium (M), High (H) scale. Figure 5.5.1 shows what the resulting deliverable DSM looks like and how it is read.

![Diagram showing deliverables and impact]

**Figure 5.5.1: Identifying inputs, lead time, perceived quality and impact in the deliverable DSM**

The deliverable DSM is a square matrix of 596 rows by 596 columns. All deliverables are listed in columns and rows in the same order, sorted first by delivery date in months before job 1 (which is embedded in the deliverable ID, see above), and then by alphabetical order of
the deliverable IDs. Using the usual DSM convention, inputs are in columns, listed on the top, and they feed to the rows, the outputs, corresponding to deliverables that require them as input. The example of figure 5.5.1 shows that deliverable DCP50c, owned by DCP (Component Engineering) and delivered at 50 months before job 1, is a required input for QUA (Quality) to produce deliverable QUA36a to be issued at 36 months before job 1. Instead of having just a mark in the corresponding cell, a chain of characters is entered to answer the three questions asked:

- **Lead time**: How long before the output delivery date (36 months before job 1) does its owner (QUA) need to receive the input (DCP50c)? In this example the answer is 12 weeks, which fits in the interval between the input reception (50 months before job 1) and the output delivery (36 months before job 1). Because many deliverables shared the same delivery date in months before job 1, in order to really identify inputs that were arriving too late, the lead time was asked in weeks and not in months. What this lead time refers to is not the value-added work time necessary to process the information of the input, but the overall time window when the team needs to have the input available before the output can be delivered. The fact that they also work simultaneously on other projects is embedded in their evaluation, and therefore depends on it, as well as on the number of engineers the team comprises. When this issue was raised, engineers responded that their headcount was fairly stable over the years as compared to the amount of work that they needed to do, and that assessing the overall lead time provided relevant enough information to analyze the process.

- **Quality**: How is the incoming quality perceived by its recipient, low in the example

- **Quality Impact**: How much the quality of the input (DCP50c) impacts the quality of the output (QUA36a), highly on the example.

The example of figure 5.5.1 shows a relationship that is perfectly fine with timing, but with an important quality issue: the input has a low perceived quality, but the impact of the input quality on the output quality is known to be high, so the output quality is very likely to be low in the eyes of its owner too. This is typically the kind of situation that the program manager was looking for when he suggested that we also ask these quality questions. All results on timing and quality will be shown in section 5.5.3.

What figure 5.5.1 does not show is that QUA36a can have several inputs, and most deliverables actually do, and that the information entered on lead time, perceived quality and quality impact is specific to each input identified. QUA36a might require an input very early because of a long lead time, and another input later because of a shorter lead time. Quality appreciations might be different between the different inputs. In the end, a considerable amount of data was needed from the process owners. An appropriate data collection process was necessary to achieve this goal in an efficient way.

### 5.5.2 Data collection process

First, the questions were asked to the process owners about their inputs rather than about their outputs, because experience in DSM research shows that people know much more accurately what their needs for information are and who to get it from, than who needs their own information. Although both approaches might seem symmetrical at first sight, a DSM built by asking for inputs (filled in rows) will be more complete and more reliable than one built by asking about outputs (filled in columns). The reason simply is that people know very
well on which information they depend (it is vital to their work), and know much less who
depends on it (because once the work is done, they do not need to worry about where the
information goes, it is not vital to them).

Contrary to the regular DSM building procedure described in Chapter 3, there was no way
the deliverable DSM could be built by interviewing each process owner and reviewing the
whole process. The potential total number of cells to review was indeed\(^2\) 595*596 = 354620,
a quantity of information clearly not manageable through interviews. In fact, from the
preceding paragraph’s point, each process owner only had to identify the inputs to the
deliverables they owned, because nobody else was in a better position to do so. The number
of cells to review by process owner \(i\) was therefore reduced to\(^3\) 595*\(x_i\), where \(x_i\) is the
number of deliverables owned by team \(i\). It was still a very large amount of data to review.
Also, after talking to several process owners, it had appeared that in some cases the lead team
had not been identified correctly and that the real lead team was listed as contributor to the
deliverable. There was no time left in the stay of the author at Ford to go through another set
of interviews with everybody to recheck the data for each deliverable (delivery date, lead
team, contributors, recipients, verbal description) before collecting the deliverable DSM data.
Also, when the deliverable was truly produced collectively by a set of contributors and no
real lead, it made sense to ask each and every contributor, on top of the team artificially
listed as lead, which inputs they needed for their contribution to the collective deliverable.
This procedure was generalized to all deliverables because it was hard to differentiate those
whose lead owner had not been identified correctly from others, and those which were the
result of teamwork from others. The potential data conflict resulting from the fact that
several different teams (the lead and the contributors) are asked to identify inputs for the
same deliverable is discussed later on in this section. The number of cells to review by
process owner \(i\) had therefore jumped to 595*(\(x_i + y_i\)), where \(x_i\) is the number of deliverables
owned by team \(i\) and \(y_i\) the number of deliverables that team \(i\) contributes to. Such
preposterous amount of data was not practically collectable systematically as such through
the regular interview process.

The author therefore decided to create electronic surveys that would help the process owners
find their relevant inputs more easily. Each deliverable was indeed expected to have only a
small number of inputs per deliverable, clearly less than 10, and it would have been
extremely long and equally annoying for the process owners to scan the complete list of 595
potential inputs just to find the three or four that were relevant to them. The number of cells
to review would be lowered to a maximum of 10*(\(x_i + y_i\)) if there was a way to filter for each
considered output the list of potential inputs to 10 or less deliverables that make sense for
this output to the team \(i\) representative. Relying on the experience that people know pretty
well what their inputs are, the author designed a procedure that filtered the list of inputs
based on the process owner’s knowledge on which teams provide him input and when.

\(^2\) As explained in chapter 3, diagonal marks are usually meaningless in DSMs, and they are indeed in this
deliverable DSM because a deliverable cannot be an input for itself. For each row, 595 columns out of the 596
can potentially be an input. There are 596 rows so the total number of cells to review is therefore 595*596.
\(^3\) Same explanation as footnote 2, but process owner \(i\) only reviews \(x\), rows.
Before describing precisely the layout and operation of the electronic survey, the author needs to mention that one more set of objectives was added to this new data collection process. The author and the research supervisor of this work were concerned that focusing only on deliverables would limit the study to formalized, official information exchange that is embedded in defined deliverables, leaving aside the fecund informal learning and sharing that goes on in meetings, phone calls, email exchanges, chats around the coffee machine, etc. The author therefore tried to enhance the study from solely focusing on official deliverables to encompassing some amount of the informal communication. The way it was done – again there was no time for a completely separate study of informal communication – was to ask lead teams whether they were receiving feedback from their recipients after the deliverable was issued, and conversely ask process owners who were identifying their inputs whether they were giving feedback to the lead team delivering them. The corollary question of whether the feedback was followed by an update was also asked in both cases.

The survey requirements had therefore become increasingly demanding for the process owners, asking them to identify their inputs, evaluate the lead time they need to process them, assess their quality, evaluate its impact on the quality of the output and adding the informal feedback and update questions. The survey design had to be intuitive and clever enough to help process owners overcome the challenge of the novelty of the method and... of the enormous amount of work required. It was a big challenge to the author, very little time was available to design the survey, test it in real conditions with real process owners, modify it and train everyone on how to use it. All process owners merit being thanked again at this point for their patience and perseverance in this process, I know many of them had very little work time available to commit for this process improvement project, and some of them spent tens of hours filling their survey, including after hours, at home and even in the weekends. Without their knowledge and the work they put into the project, this thesis just would not show much result.

The way the survey was designed is shown in figure 5.5.2.1. A specific survey was sent to each process owner, with only the data relevant to his team.

On the left, listed in rows, are the deliverables that the team owns or contributes to (not differentiated), sorted by delivery date chronologically. These rows show the deliverables that the person filling out the survey participates in, that is his team’s outputs. The deliverable ID, the sets of contributors and recipients and the verbal description of the deliverable are included on the left stub.

Across columns is the complete list of all deliverables, among which the surveyed person will identify inputs to specific deliverables in row in which they participate. The columns are sorted in three groups:

---

4 Reality is slightly modified thereafter not to confuse the reader. Because of the limitation of Microsoft™ Excel™ to 256 columns per worksheet, the total list of 596 potential inputs had to be in rows to fit in one worksheet, so the real way the surveys were designed is the transposed of all the figures presented thereafter. When assembling the data contained in the surveys, the DSM convention was restored. In order not to confuse the reader, all figures are presented with the same usual DSM convention. The final big deliverable DSM of size 596*596 had also to be split into three worksheets because of the same MS Excel limitation.
• First the deliverables that the surveyed team is already known to receive (from the recipients list of the deliverables). The surveyed team is most likely to find their inputs in this category, that’s why the corresponding deliverables were grouped together.

• Then the deliverables that the surveyed team participates in (either owns or contributes to), that is the exact same list as the one in rows. Teams can indeed reuse their own previous work as input for later.

• Finally all other deliverables not in the first two categories. Since the first step of data collection was not perfect, it was still necessary to consider this last category of potential inputs.

<table>
<thead>
<tr>
<th>Search engine controls</th>
<th>Deliverables you are reported to receive</th>
<th>Deliverables you produce &amp; contribute to</th>
<th>All other potential inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feedback questions</td>
<td>Do you receive feedback from your recipients?</td>
<td>Area where input data is entered e.g.: 12LH 596*(x_i + y_i) cells</td>
<td></td>
</tr>
<tr>
<td>Deliverable: you own &amp; contribute to (x_i + y_i rows)</td>
<td>Questions on feedback you give to your inputs' owners</td>
<td>Feedback given to the input owner?</td>
<td></td>
</tr>
<tr>
<td>Comments section</td>
<td>Your comments on any deliverable</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.5.2.1: DSM deliverable electronic survey layout for team i (not to scale)

x_i: # of deliverables owned by team i. y_i: # of deliverables contributed to by team i

On top of this general structure of the survey, three smaller sections are included:

• Questions relative to feedback received from recipients of the surveyed team’s deliverables (yellow columns). For each deliverable they participate in (i.e. each row), the surveyed team representative is asked whether the team usually receives feedback from anyone about the deliverable once it is issued (the answer is the list of teams giving feedback), and if this list is not empty then whether the team usually issues an update of the deliverable to take into account the feedback (answer is yes or no).

• Questions relative to feedback the surveyed team gives to the owner of the inputs they identify (but last bottom rows of the survey). Each time the surveyed team identifies an input in the input data area of the survey, they are asked to scroll down to the feedback questions rows at the bottom of the survey to state whether they usually give feedback to the owner of the input (answer is yes or no), and if yes then whether the owner usually sends out an update of the input to take the feedback into account (answer is yes or no).
• Comments on deliverables (last row of the survey). Surveyed teams had the opportunity to add comments on each deliverable, including their own, to complete the data, modify it or propose that a deliverable is suppressed or added.

The example rows (in green) remind the surveyed person of the format of the data they should enter (see figure 5.5.2.1).

The “Search Engine” is a background Microsoft Visual Basic code embedded in the survey Excel spreadsheet that helps the surveyed person to find his inputs among the 596 potential columns. As explained earlier, although the potential inputs across columns were sorted in three categories, some teams were known to receive a great number of deliverables (see figure 5.4.3) and scrolling through all of them in columns was too long to make the survey realistically practicable. As shown on figure 5.5.2.2, when clicking on the “search engine” button, the user is asked to estimate the delivery date of the input he or she is looking for, and the team the input is thought to belong to. Based on this information, the code filters the list of deliverables that the claimed owner participates in (owns or contributes to) to a range of ±4 months around the date entered by the user. From this filtered list the user can more easily select the deliverable he or she is looking for (the verbal description appears in the filtered list) and the code will then move the spreadsheet cursor to the corresponding column in the matrix (figure 5.5.2.3). The user then only has to fill in his input data in the cell at the intersection of this column and the row of the output he is producing from the identified input. If the deliverable is not found in the filtered list, the user can change his search parameters to try again. If the input cannot be found, the user jumps to the comment section and enters the missing data characteristics.

![Figure 5.5.2.2: Survey “search engine” control window](image)
### POTENTIAL INPUTS FOUND FOR YOUR REQUEST

<table>
<thead>
<tr>
<th>Data ID</th>
<th>Contributor</th>
<th>Recipient</th>
<th>Deliverable Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASG207</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF207</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF208</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF209</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF070</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF071</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF072</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF073</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF074</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF075</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF076</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF078</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF079</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF080</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF081</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF082</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF083</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCF084</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EPD844</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

List of potential inputs to which the chosen team participates within + or - 4 months of the entered delivery date (hidden)

---

**Figure 5.5.2.3: Survey “search engine” input selection window**

Some substantial training was provided to the users before they started filling out their surveys. A document with written instructions was given to them together with the survey (in Appendix D), and the author spent about an hour with each user to explain it and have them practice the quite complicated way the survey works. This strategy of individual training, chosen for its flexibility in the scheduling of the training meetings, did not prove efficient as many users stopped working on their survey out of confusion on how it functioned. The survey activity also represented a considerable amount of time but it was clearly not a priority to many users who had already plenty to do on their regular agenda. A special motivating event was required to finally record the missing information that would enable a fairly detailed process analysis. A meeting in one of the computer training rooms of the building was organized in late September 2002 to have all the users who had not completed their survey yet in the same room, so that they could learn from each other’s experience on the survey, all benefit from the author’s advice at the same time, and not be distracted by other tasks. This meeting was successful as many of the remaining surveys were completed, with only two very heavy ones still incomplete. By the end of November 2002 the author had received all the necessary data to begin the thorough analysis.⁵

---

⁵ Two surveys were actually never returned for analysis, the Purchasing one for reasons explained later in section 5.6.1, and the CAD one because its owner had already filled the enormous systems engineering and component engineering surveys and had no time left for the CAD one. Additionally, all functional teams outside the commodity design process (vehicle operations, suppliers, etc.) were not asked to fill a survey. These missing surveys played a small statistical role individually in the process so the results presented in section 5.5.3 are still largely valid. Overall, there are 99 deliverables out of the 596 for which the owner was not asked to identify inputs (which would for most deliverables be out of the commodity design process deliverable list anyway). However, these 99 deliverables could be identified as inputs for other deliverables.
5.5.3 Results and interpretations
Once all the surveys were received, the "sliced" DSMs needed to be reassembled into a single complete one with all deliverables both in rows and in columns. The only possible data conflict between surveys was the case when different participants to a single output (all were asked about inputs for the common output in their surveys) identified the same input with different lead times and / or quality information. In this case, two pieces of information were competing for a single cell in the assembled DSM. To enable automatic analysis, there could only be one string of information per cell. The author therefore decided to merge the data for the conflicting cells by recording in the assembled DSM the longest lead time, the worst quality perception and the highest quality impact. Therefore the worst case situation was recorded. A new 3 letter team code was created to differentiate these merged cells, MGD for merged, and the real strings of data from the individual sources were recorded as a comment to the cell to keep track of who entered what data. Out of the 2176 links between deliverables declared in the surveys, 2001 distinct cells are non empty in the deliverable DSM, only 175 non empty cells in the deliverable DSM (8% of all non empty cells) were therefore merged from several surveys’ data.

Process analysis was also done automatically, following this procedure:

1. Scan the deliverable DSM cells:
   - Each time a cell is not empty (i.e. a link input → output has been identified),
   - Check if the lead time fits in between the delivery dates of the input and the output to determine the timing status of the link. The test and the color codes used to differentiate the timing situations are shown in figure 5.5.3.1. Inputs are said late from the point of view of the output: inputs arrive too late for the output to be delivered at its announced date if there is not enough time (as compared to the declared lead time zz) to do the work in between XX and YY. Such delay is structural because it does not refer to an accidental longer lead time than planned, but rather to a schedule of input and output delivery dates that structurally does not leave enough time in between to do the work. This analysis assumes that the inputs are actually delivered at their announced delivery date and does not take the accumulation of delays into account. See below for more details on this through a critical path analysis.
   - If the quality information is ‘LH’ (Low perceived quality of the input and High impact of the input quality on the output quality), mark the deliverable by coloring its verbal description in row and column in blue. Also add it to a special list of critical quality deliverables to be reviewed later. (see Appendix E for excerpts of the real DSM)
   - Record the link information of the DSM into a more traditional table / list format in a different worksheet, to be fed later into a real database management software.
   - Update all kinds of counters to enable analyses presented further.
2. Once all the DSM cells are processed, determine the timing status of each deliverable by finding in its row the worst timing situation among its inputs. An output is delayed as soon as one of its inputs is delaying it. The same color code of figure 5.5.3.1 was used to color the deliverable ID of each row (see Appendix E). Counters were also updated.

Excerpts of the overall colored deliverable DSM are presented in Appendix E, from which several comments can be made already.

The deliverable DSM is somewhat different from all other types of DSMs built so far (see Chapter 3) because the order of the deliverables in rows and columns is governed by a delivery date and the DSM cells also contain timing related information. The precise border that the matrix diagonal draws between feed forwards and feedbacks in regular DSMs is not very relevant to this deliverable DSM. Indeed, as shown in figure 5.5.3.2, a mark in the upper diagonal, if relatively close to the diagonal, can very well be a relationship between two concurrent deliverables that are delivered at the same date. A mark further away from the diagonal in the upper triangle, however, linking an input to an output delivered earlier, has no chance of working well whatever the lead time is. Marks on the upper triangle are therefore likely to be timing-problematic links, but not automatically. On the other hand, a mark in the lower triangle below the diagonal is not necessarily a nice and clean link between deliverables, because no matter how far away the mark is from the diagonal, if the lead time for processing the input is bigger than the gap between the delivery dates, the link will cause delay to the output too. Since the focus is on scheduling, the deliverable DSM diagonal therefore does not play a very important role.
Figure 5.5.3.2: The deliverable DSM diagonal and timing issues

The notion of feedback loop and rework in the traditional DSMs is also challenged by the deliverable DSM because elements represented are deliverables and not tasks. Therefore, the same task can be represented by several deliverables (updates of the same document), e.g. the decks of assumptions updated and distributed throughout the process. What would be a feedback from component engineering teams to the assumption teams in a task DSM shows as a feed forward in the deliverable DSM since the component engineering teams’ information feeds into the next update of assumptions, not the previous ones. Clustering is therefore less meaningful because of the limited number of marks in the upper triangle (only 14% of the links).

Most marks are grouped around the diagonal, showing that recent information is mostly used. Scattered marks are found further below the diagonal, but very few marks are in the upper triangle far away from the diagonal. Figure 5.5.3.3 shows the details of the geographical distribution of the marks in the DSM. The distribution is very asymmetric around the diagonal, with much higher below the diagonal than above.

2001 relationships between deliverables were identified, yielding a Whitney ratio of 3.36. (see Chapter 3 for a definition of the ratio and a discussion of its meaning). This low result, compared to the average of 6.2 consistently found in DSMs, should be interpreted with care since the nature of the DSM is radically different. This matrix deals with relationships between documents or events rather than physical parts of a system. There is first no reason to believe that the interrelationship structure between documents should reflect the same structure as the physical system, which would be comparable to those with 6.2 connections per element on average. Secondly, Qi Dong found in her M.S. Thesis (Dong, 1998, p.117) that only 30% of the system-level knowledge was usually recorded in documents, as opposed to 70% found through interviews with engineers. Even if document
interrelationships and system structure were linked, a large part of the links would still be missed, which could start explaining this low result.

**Figure 5.5.3.3:** 1. Geographical distribution of marks in the deliverable DSM (not to scale) 2. Density of the matrix (average number of links per cell, to scale)

The 7 plotted dots of the curve in figure 5.5.3.3 represent the density of marks defined by the number of marks over the number of cells of the identified zone of the DSM. For example, in zone 1, a right isoceles triangle of side 537, there are 627 marked cells (links) out of a total of 144453, yielding a ratio of 0.0043. The blue curve is an extrapolation from the 7 dots.

The distribution of timing status results is shown in figure 5.5.3.4.

**Figure 5.5.3.4:** Timing status of links

Obviously, the above results are highly dependent on the delivery dates declared at the initial give and get onsite meeting, and on the lead times entered in the surveys. But both sets of

---

6 Links on the diagonal are irrelevant, they are probably mistakes made by survey users. Because their number is negligible, they do not play any role in the rest of the analysis.

7 Number of cells in zone 1 is \( \sum_{k=1}^{537} k = \frac{1}{2} (537 * 538) = 144453 \)
data were consistent since the same people were interviewed at both data collection stages. The distribution of links, out of a total of 2001, therefore yields the following interpretations.

- 12% of relationships between deliverables induce a serious delay in the delivery of the output by more than a month. This figure is relatively small compared to the general appraisal of the situation made by the practitioners and described in section 5.1.3. Analysis of the deliverables timing status will explain it better. These long delayed links are likely to be quite difficult to fix because more than a month of schedule shift usually involves some important schedule shift in the upstream flow of deliverables, that is potentially creating new important timing disconnects.

- 18% of relationships induce a minor delay in the delivery of the output, probably easily fixable since most teams can probably handle a two to three week shift in their schedule with little disturbance. The total number of late input → output relationships however amounts to 30% of the total number of links, which is very significant.

- Only 22% of the links are just-in-time, which is the best information situation (the team that receives the input gets the most recent information available) but also the most risky one (if the input is available later than planned for some reason, or if the input needs to be updated after its delivery date, the output team has no slack time and the output is automatically delayed).

- 48% of the links exhibit some time slack, which gives nice process flexibility on the one hand, but involves using out of date information if the gap is too long on the other hand. At least these links do not create delays in the process, but they leave room for quality improvement (using more recent information) and reducing time slack or hand-offs.

The timing status of a deliverable is then defined by the worst timing status among the links from its inputs. In other words, if a deliverable has for example four inputs, two with timing slack links, one with just-in-time link and one with a link bearing a timing disconnect of less than one month, the deliverable will be delayed by less than one month and that will be its timing status. The distribution of timing status among deliverables is shown in figure 5.5.3.5.

![Pie chart showing timing status of deliverables]

**Figure 5.5.3.5: Timing status of deliverables**

The distribution of deliverables per timing status is somewhat different from that of links because of the domination of delaying links over on-time and time-slack ones. It is important to stress again that the following discussion, just like the pie charts of figures 5.5.3.4 and 5, assumes that inputs are available at their delivery date and that the work to produce the output starts on time. It only focuses on whether the gap between the delivery dates of the
input and the output leave enough time for the work to be completed in between (as compared to the lead time entered in the matrix). The fact that these inputs might themselves not be available at their declared delivery date because of upstream timing disconnects is not taken into account. In fact, if the accumulation of delays is taken into account, based on fixed “root” deliverable dates and without computing any iterations between deliverables, job 1 would happen 18 months after schedule. Reality is of course different, lead times are shrunk by management pressure to go quicker and quality or targets are sometimes a little bit compromised, but on the other hand it is very common for programs to delay their job 1 by several months as they find out that they are not going to meet the deadline or if a they run into a major issue. It has happened several times in a row for Prog, the particular commodity program studied. The calculation above supports the idea that these delays might not only be due to unexpected engineering problems, but actually to the design process itself.

- The 12% of long delaying information flows (links) drive 22% of the total number of deliverables to be delayed by more than one month. Again, this happens because there needs to be only one long delaying link to a deliverable to delay it by more than one month, other categories of links are hence dominated, leading to a redistribution of the timing status increasing the delay proportion and decreasing the time slack one. 22% of deliverables delayed by more than one month is now a serious portion of the process, but it is good to know that “only” 12% of the links (i.e. 240 links) need to be fixed to resolve the issues (provided that fixing one link does not create more trouble elsewhere).

- Another 18% of the deliverables are delayed by less than a month because there is not enough time to work on the inputs once they are received. That makes a total of 40% of delayed deliverables in the process, without taking into account the accumulation of delays...

- 17% of deliverables can be issued just-in-time based on the announced delivery date of their inputs. These deliverables do not leave room for any margin of error in scheduling.

- 19% of the deliverables exhibit a timing slack which can be taken advantage of by allowing either inputs to be delivered later or the deliverable to be completed earlier. The latter change would nevertheless make the process shorter only if the deliverable is part of the critical path. On the other hand, these deliverables constitute the flexibility buffer of the process and allow some change in the schedule without consequence. Unless the slack is really big, it might be a good idea to keep such flexibility. The question is how much.

- Finally, 23% of the deliverables (137 out of 596) had no predecessors and apparently did not require any input. This is an important number of deliverables. They are either the initial deliverables to start the program (generational learning between successive programs is not included in the study), deliverables for which inputs were not part of the deliverable list (e.g. inputs from outside the commodity division, or too technical and too detailed to be in the list), or events and not documents. Work needs to be done on

---

8 This statement is totally equivalent to a saying that the critical path length of the process is 60+18 = 78 months
9 So as to get more recent information and hopefully better quality of information if applicable. See the discussion below on low quality high impact deliverables for more details on this subject
10 Events were often overlooked by subprocess owners filling their surveys, because they do not convey information by themselves: a sign off event is not used later on, but the document signed off is. However, there should be connections between deliverables and events, in the example of a sign off, the deliverable should be an input to the event.
these deliverables to find out why they do not have predecessors in the process, whether it is normal or some data is missing. Details about the way this work is implemented are given in section 5.7.

Looking the other way around at the deliverable interrelationships, i.e. where the deliverables feed to, 143 deliverables (24%) were found not to feed anywhere. These deliverables can be the final ones, like job 1 or final reports used for the next generation of commodity, or events that do not technically serve as inputs for later deliverables. An impressive number of deliverables that do not feed anywhere are however purely design ones, and the way the deliverable list had been constructed made so that the deliverables they should be inputs for were not represented. This phenomenon relates to one of the explanations of the unmatched gives and gets: the inconsistency in scope used by different teams in the precision of their deliverables. To evaluate the extent to which events introduce a bias in the figures, the number of deliverables that both do not have predecessors and do not feed anywhere, which are likely to be simple events, was counted. 34 deliverables corresponded to that description. It seems therefore that some of the 109 deliverables that do not feed anywhere and some of the 103 deliverables that have no predecessors, together accounting for about 36% of the deliverables, give evidence that some data is still missing in the deliverable DSM.

![Diagram]

**Figure 5.5.3.6: Intersection between deliverables that have no predecessors and those that do not feed into any other**

The inputs that cause serious delay to their output (because they do not leave enough time between the delivery dates, more than one month short) are quite varied. In the early phase of the process, many important directions and decisions fall in this category: initial design directions, initial cost estimate, etc. Technical deliverables also pose timing problems from the beginning, initial targets are given too late to the CAE analytical teams for them to produce their required outputs feeding the design teams. There are delaying relationships between the CAD team and the component engineering teams too. Sourcing decisions also seem to be problematic for the coordination with suppliers, in some cases the potential list of suppliers is given too late to the STA (supplier technical assistance) team that is in charge of evaluating their process quality. As the process moves on, the build process poses problems and they reappear consistently on each generation of prototype: the design freeze is decided too late for the component engineers to provide the machining and the inspection data to the prototyping shop. Testing results also exhibit some timing problems, in connection with
quality documents. Some other timing disconnects of more than one month also look very strange and are probably mistakes that will need to be fixed by engineers.

From the quality point of view, 80 deliverables with low perceived quality and high quality impact were also identified. As explained earlier, the quality that is being dealt with here is the information quality, not the product quality. A deliverable is qualified as a low quality one if it is not considered to contain the final information it is supposed to convey, if its information is questionable, not very reliable, if there is enough information to fulfill the needs of the teams that use it as input. The flagged deliverables are mainly early deliverables, assumptions, targets and planning deliverables, that is all the directions given to the designers at the start of the program and updated later. A number of criticized deliverables also come from outside the program, for example from teams in charge of interfaces with the vehicle. Very few design deliverables are criticized, except a design freeze event for a prototype and some component quality documents. As explained earlier, Prog had gone through a series of major architectural changes in its beginning, and some frustration over management decisions had grown in the process participants. The data is therefore biased by the particular history of Prog. However, when reviewing these results with former Ford high-level decision makers, one of them mentioned that such changes are common in programs that intend to do everything at the same time, that is to address a large number of different applications in the same program, the case of Prog. By trying to accomplish many different goals, they draw competing pressures from different managers on them, thus the changes in strategy, and sometimes these programs end up being canceled. The data therefore seems to be representative of a class of these major programs with unreasonable expectations. Table 5.5.3.7 evaluates how much timing issues were taken into account in the assessments of low quality by looking at the overlaps. It turns out that there is clearly no correlation between late information flows and the low quality high impact assessment. When subprocess owners qualified an input as ‘LH’, they did not take into account the fact that it was potentially delayed but only the quality of the information, as they were requested to. The data on critical quality deliverables is therefore reliable for interpretations on quality. Table 5.5.3.7 actually conveys the idea that most low quality high impact statements apply to inputs that are used much later than their delivery date. It might be useful to update them to reflect changes since the initial version before they are used for later deliverables. One can think that their perceived quality would improve this way. For just-in-time links with low quality assessments, the second largest group, only the content quality can be improved, unless the delivery date is changed to improve quality and thereby the timing status too.

<table>
<thead>
<tr>
<th>Among links with LH assessment:</th>
<th># of links, %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Links with timing disconnect &gt; 1 month</td>
<td>21, 6%</td>
</tr>
<tr>
<td>Links with timing disconnect ≤ 1 month</td>
<td>31, 8%</td>
</tr>
<tr>
<td>Just-in-time links</td>
<td>85, 23%</td>
</tr>
<tr>
<td>Links with timing opportunity</td>
<td>231, 63%</td>
</tr>
<tr>
<td>Total</td>
<td>368, 100%</td>
</tr>
</tbody>
</table>

Table 5.5.3.7: Data proving that the low quality high impact assessment is decorrelated from the lack of time to process the input
Table 5.5.3.7 shows 368 links (out of the 2176 links declared in surveys) between deliverables identified with a low quality high impact statement, although 80 deliverables are flagged with this assessment. This is because one deliverable can be criticized several times if it feeds into more than one deliverable or if several contributors declare it as low quality input for the same output in their different surveys. In fact, the number of times a deliverable is criticized was taken into account when the data was reviewed later by subprocess owners. Details, also on deliverables with no predecessors and that do not feed anywhere, are given in section 5.7, “task force next steps”, after organizational aspects, in particular on the low quality perception, are reviewed in section 5.6.

Finally, to analyze the early phase of the process, thought to be more problematic than the later phases; a study of the growth over time of each metric described above was conducted. Results appear in figures 5.5.3.8 and 9. Before looking at the deliverables’ timing status, which is what really matters for the process progress, looking at links’ timing status enables to understand what the deliverables’ timing status is driven by. The chart of figure 5.5.3.8 is designed to track the speed of growth of each number of links in its own category. If a certain type of link is more frequent in the early phase of the process than later on, it reaches higher levels quicker than other types. The number of timing disconnects links of more than one month actually grows slower than other categories. Timing disconnects of less than a month do occur a lot at the early phase of the process (more than one third of all these short timing disconnects have already occurred by month 56). The process also seems to be significantly driven by just in time deliveries between months 28 and 23. Most strikingly, the number of links including a critical quality judgement (Low perceived quality and high impact) is extremely important at the very beginning of the process (especially at 56 months before job 1) and almost inexistent later on. This largely confirms the perception of problematic early phase of the process. The reason why the quality criticism is so high at 56 months before job 1 is because a lot of deliverables are produced to give initial directions to the program, and each of these deliverables is criticized by all teams who use it as inputs, and all criticisms add up in the curve of figure 5.5.3.8. This multiplication of criticisms is removed in figure 5.5.3.9, in which the overall status of the deliverable is counted, independently of the number of times it was criticized. Table 5.5.3.7 gave another reason why the early deliverables bear so much criticism: they are reused much later, when they are already obsolete, so their quality is then judged unsatisfactory.

![Figure 5.5.3.8: Cumulative normalized number of links by timing status over time](image-url)
Figure 5.5.3.9: Cumulative normalized number of deliverables by category over time

The chart of figure 5.5.3.9 illustrates the speed of growth of each category of deliverables (the number in each category is normalized to the total number of deliverable of this same category over the process). Deliverables with timing disconnect of less than a month do grow faster than other categories at months 56 and 34, but deliverables with timing disconnects of more than one month exhibit a less significant similar pattern. Overall, timing disconnects are a little more numerous in the early phase than later, but not very significantly. This is the reason why just-in-time deliveries and timing opportunities grow slower than other categories although figure 5.5.3.4 showed that they were the majority of links, they are dominated by late links. However, it is very clear that critical quality deliverables (low perceived quality and high quality impact on the output) are much more present at the early phase of the process than later on. 30% of the total number of critical quality deliverables have already been issued by month 56, and 54% by month 50, to be compared with 12% for just-in-time deliverables at the same date. Two different interpretations of this phenomenon are possible.

1. From the discussions the author had with the process participants and from the history of Prog, this result can be seen as a confirmation of people’s frustration over the early directions and strategic decisions made, that kept changing. This conclusion is based on the assumption that critical quality deliverables (low quality, high impact) should not grow any faster than other categories of deliverables.

2. It can also be argued that a higher concentration of critical quality deliverables in the early phase of the process is normal and even a sign of good process health. The commodity is not designed yet, and there is a lot of uncertainty in each deliverable process participants produce, thus the low assessments of deliverable quality. Additionally, if the low perceived quality of deliverables reflects the identification of problems, it is better to identify them in the early phase when they can be fixed than later. Such conclusions are based on the assumption that critical quality deliverables should be more present in the beginning, that it is both normal and desirable.

From the author’s point of view, both interpretations apply to the data. Undoubtedly, process participants do not like the uncertainty of the beginning of a process, and uncertainty creates distrust in the information exchanges. But undeniably this phenomenon was amplified in the
data by the lousy history of the beginning of Prog's process. The issues of the early phase of the process are therefore confirmed, including a higher proportion of deliverables late by less than one month, but most strikingly a very strong proportion of deliverables criticized for their low quality by numerous teams. The extent of the problem is hard to evaluate at this point because of the conflicting influence of both interpretations described above.

5.5.4 Conclusions so far
It is useful to back up to review the accomplishments of the case toward the thesis goals.

Process
The commodity design process was mapped in great detail, encompassing the information flows and interrelationships between deliverables, which are key for scheduling and process-level understanding. Unlike FPDS, the commodity deliverable DSM is a process, not just a schedule, because these relationships are captured, and the drivers of each task are understood. The process was analyzed to identify timing and deliverable quality issues, which are fully ready to be addressed (in section 5.7).

Project
The deliverable DSM also gives clear roadmaps of what work needs to be done and when between milestones. Project progress can be tracked along the diagonal of the DSM, consequences of punctual delays and / or problems can be anticipated by looking at the downstream of deliverables driven by the problematic one. Special focus can be given to deliverables that are known to be unsatisfactory to their recipients. The accomplishments in terms of project management are close to what a project management software like Microsoft Project™ could do. However, the DSM approach provided more flexibility in the formatting of the data (allowing quality assessments) and above all a much easier way of entering and visualizing the complex and varied connections between deliverables. Since much of the structural benefits of DSMs were lost because of the nature of the deliverable DSM, it is however true that the difference with project management software tools is rather limited.

Knowledge
The process is documented in a format that can easily be used as a reference by anyone. Process-level knowledge is captured and shared through the deliverable DSM or excerpts of it (see section 5.7). Teams have also learnt a lot about how their work is used and perceived by other teams, and what the data says is their general role in the process, with which proportion of ownership, contribution and reception of deliverables. The results of this work pattern analysis (section 5.4) can be surprising for some teams, and they can learn a lot from trying to understand why the statistics differ from their own understanding of their role and working on reconciling the two visions.

Organizations
Apart from the analysis of the involvement of each team in the process and the new understanding of how other teams use their work, little has been done so far to aggregate the data enough to understand the structural relationships between functional teams. While process mapping, project management and knowledge management have effectively been performed concurrently using the deliverable DSM, organizations management has been left
out so far. So has the issue of informal communication through feedback. That is what the next section deals with.

5.6 Organizations DSM: enabling a dynamic management of organizations

The extremely large quantity of data available from the description of deliverables and their connections in the deliverable DSM enable the analysis of many different parameters and metrics. In particular, it is fairly easy to aggregate the detailed data from deliverables to quantify information exchange between teams overall in the format of organizations DSMs. However, it is interesting to start with more qualitative observations of organizational processes during the case study.

5.6.1 Observations on functional organizations’ cultural disconnects

The demanding requirements of the task force to fill the survey occasioned very interesting observations of cultural differences and tensions across functional teams. In the most obvious way, the resistance that two teams, Purchasing and Manufacturing Engineering, opposed to committing the necessary time to fill out the surveys, was an excellent illustration of the phenomenon. First of all, they were obviously very busy, but so was everybody else. These teams’ representatives, very nice and friendly apart from this particular situation, put forward arguments to defend their unwillingness to spend the necessary time on the surveys that, in the author’s view, illustrate the cultural disconnects across different functional teams presented in the introduction chapter as an important issue in PD processes. The analysis that follows should therefore not be understood in any way as a personal criticism to the particular purchasing or manufacturing teams’ members of Prog, but rather as a big picture thought on old cultural differences between manufacturing and design on the one hand, and purchasing (or more generally management functions) and design on the other hand. Here is a set of quotes from a Manufacturing engineer (who had been the nicest person possible, so again it has nothing to do with personality or personal hostility toward the author):

“Manufacturing Engineering has always had precise and well defined process and schedule, which is not the case for Product. We have no time to spend on a project from Product to define a new process because we already have a process that works well and that they don’t want to use. Manufacturing doesn’t have problems with other programs, just [Prog]. We are also working on changing our timing from [xx] months to [yy] months, so we don’t want to put several hours on a project that will never be used again. All those MBJ1 dates are subject to change. You got a very unfair assignment, I understand you want to complete it, but we spend too much time on process building and not enough on planning. We also don’t want to work on a [S6 / P6] program specific study, we want a process common to all projects.”

It comprises a mix of many interesting statements that are reviewed below.

- ‘A process improvement project run by Product’ (i.e. Design teams): it is true that the task force had been initiated and led by one of Prog’s supervisors, from the systems engineering design team. Although the Manufacturing Engineering team representative was technically a member of the task force, he never participated in the meetings, either because he did not want to be involved or because the task force supervisor had told him...
it was ok not to be involved. The task force leadership had not done the best job indeed in
getting everybody informed and motivated about its work, and on several occasions some
important participants were not even informed about key offsite meetings, like the one in
September 2002 to continue filling out surveys, and the one in January 2003 to present
results and enter a new phase of data review and real process improvement. It is therefore
not very surprising that some functions felt they had been left out of the task force work
and saw the project as primarily one of the design teams. These coordination problems
might also be a sign of the design teams’ lack of interest or implicit overlook of the
manufacturing engineers’ and others’ opinion.

• ‘My process is good and your process is bad’: this is a typical example of “fortress”
functional thinking that was described in the introduction chapter and in process
engineering section of chapter 2. In spite of the faulty involvement of some functions in
the task force, undeniably everybody knew that its purpose was to align functional
subprocesses altogether to build a coherent end-to-end process and put an end to the
frustration induced by delays and misunderstandings between different functional teams.
The Manufacturing engineer’s reaction proves that the task force was not anywhere close
to reaching this goal of everybody working harmoniously together. This engineer clearly
stated his preference for a clear subprocess of his own, that Manufacturing Engineering
can keep under control, rather than compromising with all others, especially the design
teams, to create an overall successful cross-functional process. This is the main
conclusion to take away from his point.

• ‘Prog has problems whereas other programs don’t.’ A lot was said already about the
problematic history of Prog especially at its beginning. From the author’s point of view
(influenced by discussions with experienced engineers at Ford and former Ford
executives), Prog has no more no less problems (but still many and major) than any large
S6 / P6 program that targets a lot of different architectural applications. The statement by
the Manufacturing engineer probably refers to less ambitious programs, which occur
much more frequently than large ones like Prog.

• ‘Delivery dates are doomed to change, this process project is useless’: it is totally true
that the timing of the process is highly volatile as the overall PD time keeps being shrunk
by company managers. The main point of the project was however to understand what
the process is and how it functions, mainly through the deliverable DSM. Putting a
timing label on deliverables and evaluating lead times was necessary to understand what
the limits of reducing PD time are. What really is the key of the analysis is the
connections between deliverables (inputs / outputs), which do not change much even
though the timing can possibly change. The project is therefore not at all useless, it helps
understand how much timing can be changed, and the deliverable structure will remain
even if the schedule is changed: the commodity still had to be designed…

• ‘We spend too much time on process building and not enough on planning’. The
author himself was very surprised with the number of process improvement projects at
Ford in the past (most of them having failed to drive any change) and present. The task
force could therefore legitimately be thought of as one more of these process projects.
However, the task force had two major advantages that could help it succeed: it was
under the control of the actual engineers who do the work, not a support team from
outside the process, and instead of thinking the process for everybody else in a top-down
approach, the task force had chosen to gather data in a bottom-up manner through the
give and get exercise and later on the deliverable DSM. The suggestion that time spent on process had better be spent on planning does actually not hold much in the face of the results presented in sections 5.3, 4 and 5, which prove that planning was literally impossible when there is such misalignment between functional teams and when the input and output flows show such problematic timing disconnects.

- ‘It is not enough to work on S6 / P6 process improvement, it is too limited’: it is totally true that the study being based on Prog, the data represents a S6 / P6 process and timing. However, S6 / P6 programs are the most complex and the most difficult ones, and comprise all activities that lighter programs for the commodity may comprise. Lighter programs might not exactly follow the same process as the one identified for Prog, but lessons learned on S6 / P6 process are very likely to be easily transferable to lighter processes. One can imagine that lighter processes might be roughly obtained by eliminating some tasks and shortening some others from Prog’s process. That is exactly why the commodity division quality office was aiming to use the task force work as a best practice for the entire division, not just the commodity.

The key point of quoting and analyzing this talk is to show how much cultural and political factors can influence people’s reactions. In this case, distrust between Manufacturing Engineering and Design appeared clearly and to a certain extent impaired the goal that all should have shared of unifying and aligning the process across functions. To be fair with the Manufacturing Engineering team, the survey was filled out in the end, among one of the fastest returned to the author actually, so the opposition was only initial and temporary.

Similar issues came up with the Purchasing team representative, who claimed among other things that he had not been involved enough by the task force since the beginning of the work to fully understand it. It was true to some extent indeed, implying again that the Design teams in charge of the task force also had a symmetrical bias, even unconscious, against Purchasing, Manufacturing Engineering or some others. More interestingly, the purchasing team representative claimed that the deliverable DSM survey was a lot of work that his hierarchy was not aware of and would not value: he therefore had no incentive to make it a priority over his purely purchasing work. Here is again a typical behavior induced by the excessive functional segmentation of the organizational structure. The Purchasing engineer, our example here but it could be anyone else, thinks more in terms of his home department than in terms of the program he is working for... Collaboration is therefore very difficult when participants all have hidden goals that actually compete rather than create emulation.

Many more stories and observations could be told about cultural tensions between functional teams (e.g. between a CAE team and a design team, or conversely the lack of tension in one past program for which all functions were reporting to the same hierarchy directly), or about temptation to free-ride for one’s own functional team when it impairs the common goal\textsuperscript{11}. A

\textsuperscript{11} One CAE engineer and one testing engineer changed the dates and descriptions of deliverables in rows and columns of their survey DSM, although they knew that all other teams were simultaneously working on the same common base as the one they had been provided. Any modification was explicitly requested to be entered in the comments section. They introduced bias in the data by choosing their own referential to answer their input questions although the initial give and get onsite meeting had been designed to consense on a list of...
project like the task force is already a good start to try to align all functional teams to the same goal of making an end-to-end process easier, but the reader will be convinced from the stories above that a specific focus on organizational structure was necessary. Design structure Matrices applied to organizations can also help this approach.

### 5.6.2 Organizations DSMs

The first available set of data was that of the cards / deliverables. By simply studying the relationships between the deliverable owner and the contributors and recipients for each deliverable, it was possible to get a first understanding of the communication patterns between teams. In order to quantify the relationships between teams, an information exchange "model" was necessary. Figure 5.6.1.1 shows the simplistic model that was chosen for this analysis.

![Figure 5.6.1.1: Information exchange 'model' to quantify communication between teams](image)

By simply adding all the information exchanges recorded on the deliverable data between each pair of teams (without making any difference between contributing flows and transmission / reception flows, weighing them differently is therefore a potential future work), a first organization DSM was constructed. However, this DSM (further referred to as DSM1) did not take into account the flows recorded later in the deliverable DSM and in the feedback information also asked in the surveys. (see figure 5.5.2.1). Flows recorded in the deliverable DSM were therefore compared with these already included in DSM1, and only the new ones were added. Out of the 2160 links (that can be called 'information flows' in this section since the focus is on organizations) declared by subprocess owners in their surveys, 718 were bringing new information not recorded in the deliverable characteristics (list of contributors, list of recipients) that was used to build DSM1. That is, apart from recording key relationships between deliverables to understand the process, 33% of the links in the deliverable DSM also contained new information exchange data. The corresponding new recipients were added to the list of recipients of the corresponding deliverables. The feedback and update information exchanges were also entirely added: the feedback data does not intersect with the data recorded earlier, because it deals explicitly with information exchanges that take place after the delivery of the first version of deliverables. Information exchange data recorded in DSM1 and in the deliverable DSM deals with exchanges before deliverables common to all. These two engineers therefore preferred the comfort of working in their own team's base, to trying to cope with issues all together.
and on the occasion of the delivery of the first version of deliverables. DSM1 and the additional information flows from the deliverable DSM amounted to a total of 2862 information flows, and the feedback and update information flows to 3791. This proves the importance of information exchanges outside the formal context of working collectively on delivering information and distributing it. Feedback and discussion over already issued deliverables plays a key role in the information exchange that had not been recorded before asking the feedback questions. The total resulting matrix, holding 6653 information flows, is shown in figure 5.6.2.2.

![Diagram](image)

**Figure 5.6.2.2: Overall Organization DSM of the total process**

The color code used to visualize the intensity of the information exchange between teams goes from white to light green to darker green to brown to dark red to purple and to red as the relationship between two teams involves more information exchanges. The names of teams represented in rows and columns cannot be shown for confidentiality purposes. Functions that the teams represent can however be mentioned.

One column clearly stands out in the middle of the DSM: it is the Systems Engineering team, which sends out information to almost everybody and intensively. The teams that send a lot of information (columns) to other teams (rows) are from left to right on the matrix: the assumptions team (dark green and brown cells), the component engineering team (with red cell), the systems engineering team, the manufacturing engineering team (with purple cell) and the planning team (with purple and dark red cells).

Conversely, only two rows stand out as their corresponding team receives a lot of information from other teams: from top to down the component engineering team and the system engineering team. These two teams therefore prove to send out a lot of information and to receive a lot of information: their role as knowledge hubs in the organization is clear.
Detailed information exchange therefore confirms the conclusions that derived from the analysis of deliverable statistics (section 5.4).

Clustering this overall organization DSM yields disappointing results: as shown on figure 5.6.2.3, the clustered DSM make a block of 6 teams appear: the systems engineering, the component engineering, the manufacturing engineering, the assumptions, the planning and the CAD teams.

![DSM Clustered](image)

**Figure 5.6.2.3: Overall Organizations DSM clustered**

There is therefore not much learnt in the clustering of the overall organizations DSM, since the functional teams afore mentioned are very well known for exchanging a lot of information and being required to collaborate intensively for the success of product development processes. An important reason for this absence of interesting conclusion is the fact that the data covers a period of more than five years. A finer analysis in each phase of the process follows to take advantage of the enormous amount of data available. It yields to a proposal of dynamic management of organizations.

### 5.6.3 Dynamic Management of Organizations

Figures 5.6.3.1 to 4 exhibit the unclustered organizations DSMs restricted to the information exchanged within a process phase between two major FPDS milestones. Names of the milestones and dates in months before job 1 are provided. The DSMs are unclustered so as to enable the reader to see the evolution (the order of rows and columns remains the same for all 5 matrices).

Different patterns of interactions clearly appear in each different phase, allowing different clustering of teams in different phases: this supports the idea that organizations management should adapt to the evolution of information exchange patterns of the process.
In the first phase of figure 5.6.3.1, Assumptions, Planning and Systems Engineering play the key role in distributing information. The rest of the organization is fairly empty, they are the three only teams that are already active in this early phase of the process.

In the second phase of figure 5.6.3.2, as the project really gets kicked off, everybody gets involved (as shown by the scattered marks all over the DSM). The key teams sending information are components, system engineering, manufacturing and planning.

The third phase shown in figure 5.6.3.3 is still a busy phase for everyone. However, the pattern of information exchange is somewhat different as the key teams are now component engineering, system engineering, quality teams and prototype build team, as the first prototype is being built and tested in this phase. Also, components and systems engineering do not show so much as information senders as before, but more as information receivers (colored rows appear more than colored columns, as opposed to the previous phase).
In the fourth phase of figure 5.6.3.4, many teams are almost done with their involvement in the process (e.g. assumptions) but teams like system engineering, testing, prototype build and teams outside of the system (vehicle interfaces) remain busy or take more importance.

In the last phase of figure 5.6.3.5, between the last prototype and commercial production, systems engineering and manufacturing are the only ones still really sending information out.

From this description of phases, one can therefore cluster the organization DSMs (which are not shown here because without team names they would not mean anything to the reader) and come up with phase-specific organizational enablers to facilitate the collaboration of teams that exchange a lot of information within the phase. This approach has much more value than the static vision that the overall process organization DSM was providing. A dynamic management of organizations, that really follows the process as it evolves, is suggested.

5.6.4 Who criticizes whom
To conclude with this section on understanding organizational interactions, data telling which teams make critical statements on the deliverables of which other teams can be easily extracted from the deliverable DSM. Still in the format of an organization DSM, figure 5.6.4 reads as a team in column criticizing a team in row the number of times shown in the cell. The same color code is used as previously, from white to greens to brown, purple and red as criticism gets more frequent.

![Figure 5.6.4: Who criticizes whom organization DSM](image)

Some interesting results derive from this matrix: one team is consistently criticized by many others, it is the Planning team, which bears most of the early phase of the process criticisms. It is the only team that is widely criticized by many others. Strangely enough, the assumptions team, the second row and column, is criticized by almost nobody except one team, a CAE team, that criticizes it overwhelmingly (red cell with 177 criticisms). This same
team is also the one that criticizes the planning team most (red cell with 45 criticisms). It could therefore be either a tendency to criticize more than others or a truly high dependence of this CAE team on early documents issued by the planning and assumptions teams. A meeting between this CAE team representative and the planning and assumptions subprocess owners would therefore be advisable.

5.7 Process improvement project next steps

This last section briefly outlines the steps that the task force initiated and should continue to perform after the completion of the analysis presented above.

A new code is ready to take into account modifications made directly on the deliverable DSM, whether in the links between deliverables or in the description of the deliverables’ attributes themselves. Surveys, which were still the main source of data whenever a modification (e.g. an improvement) was to be entered in the database, can now be discarded and it is much easier to make modifications on a single end-to-end process document like the deliverable DSM.

At another one-day offsite meeting in January 2003, in which the author presented to the task force most of the findings explained in this chapter, subprocess owners were provided excerpts of the deliverable DSM relevant to them. They were actually provided two views of their subprocess in the formats of matrices: the input view, with their deliverables in rows and only their declared inputs shown in columns\textsuperscript{12}, and the output view, with their deliverables in columns and the downstream deliverables in rows, containing the information as to whether they deliver their work too late for their customers or the quality of their work is perceived insufficient. This last view was totally new to subprocess owners since it resulted entirely from the surveys of other teams. The structure of these two views is clarified in figure 5.7.1. Based on both views, teams know what issues they have and what issues they cause, they can therefore little by little agree with other teams on resolutions for timing and quality alignment, and record the new data in the deliverable DSM. This is a long term and challenging task that has just started.

\textsuperscript{12} This view was almost identical to the survey they had filled, with little modifications due to the data overlap between surveys filled by contributors of the same deliverable. However, the visual aspect of this version provided a much better map of the inputs of teams than their survey because all the other non relevant columns had been removed.
Figure 5.7.1: Both views provided to subprocess owners after completion of the assembly of the deliverable DSM from surveys (matrices are not square)

Additionally, deliverables without predecessors and conversely not feeding to any other deliverable were also listed and shown to subprocess owners, in order to either complete any missing data or to confirm that it is their nature. In this last situation, subprocess owners need to ask themselves why it is so, if a deliverable that does not feed into any other deliverable is useless or not, etc.

Critical quality deliverables (with low perceived quality and high quality impact on the downstream deliverables) also need to be discussed. Why are they criticized? Is it surprising or is it a well known problem? What can be done to attempt to resolve it? Critical quality deliverables were actually reviewed in two different batches at this January 2003 analysis results offsite: the deliverables that had been criticized more than 5 times (23 deliverables) were reviewed collectively since they seemed to be problematic for many different people, whereas those criticized fewer times were left for small group meetings. The offsite of January 2003 was just a starting point, and it will take a long time to resolve them all as part of the task force weekly meetings. Deliverables with low perceived quality and medium impact should also be addressed when the high impact ones are all dealt with.

The issue of keeping events as deliverables in the database is controversial. On the one hand, they do not contain information and therefore introduce bias in the statistics of information exchange. Additionally, they were usually written on cards to reflect the FPDS milestone events that do not align with the real process, one can therefore think that they contradict the purpose of the database to record process the way it is executed. However on the other hand, these events are part of the process indeed, because it includes reviews, approvals, sign-offs, that are necessary and important. Whether these events are scheduled to align with the real work schedule is a different issue. But the author thinks they should be left in the data (if they are truly part of the process, driven by upstream deliverables and driving downstream deliverables) in order to reflect a non-negligible part of any process: decision-making. The events included in the data, just like all deliverables actually, should therefore be reviewed by an experienced engineer, for example the Prog supervisor would is leading the task force, in order to decide whether each of them individually adds value to the description of the
process or not. It will take hundreds of hours, and the question again is: can the division afford in the short term to have its most experienced system engineer work on process rather than the product?

Once all the above steps have been completed and the data is credible enough to reflect reality, and improved enough to bear limited structural delays if any, a more elaborated analysis on the critical path of the process can be made. Identifying the timing bottlenecks of the process (i.e. the longest sequence of deliverables that drives the completion date of the overall process) helps focus on issuing the relevant deliverables on time so as to keep up with the process schedule. Such an analysis cannot be done on the current map because it bears too many structural delays, and many delivery dates and connections between deliverables will be modified as the task force continues its work. Such changes will deeply impact the critical path of the process so there is no point analyzing it at this point.

The current map of the process through the deliverable DSM is also probably not satisfactory enough to begin to think about scheduling reviews and decisions at the right time to match process progress, especially prototype building and testing schedule. Indeed, with the data yielding job one 18 months after schedule without taking into account loops and iterations between deliverables, the map is not credible enough to entail serious reengineering yet. But when the deliverable DSM is sufficiently updated to reflect alignment and improvement efforts, the question will arise: if FPDS milestones are not relevant to the process, when should they be scheduled at the commodity level, and how can a new schedule both follow the commodity process and support sufficiently closely the vehicle process? The general idea, sometimes called “making justified decisions”, an expression used by Tony Zambito at Ford, is to have a milestone only when all the necessary information is available, and not beforehand. In a task DSM, such appropriate moments for justified decisions are at the lower right corner of blocks of iteration (i.e. clusters of tasks). Such moments are not as easy to determine from the deliverable DSM because of the virtual absence of feedbacks and the practical impossibility to cluster. However, with sufficient knowledge on the process, experienced engineers should be able to identify these moments from the sequence of deliverables. The issue is not only one of interface between the commodity process and the vehicle one, there are also internal issues that the author was very surprised to discover. One of the most striking is the availability of prototype testing results: full testing results for prototype batch n are only available after the design of prototype batch n+1 is already frozen. The learning from testing is therefore offset by one generation of prototype, which in a set of three batches is pretty disturbing... This issue was known by project managers, and the reaction they had when the author brought it up was “yes, we know, it’s like that, you can’t do anything about it”. Partial results are available before the design of the next generation of prototypes gets frozen, but it is still puzzling to know that full results for prototype generation n will only be useful to prototype generation n+2 if it exists...

---

13 Delays resulting from the planning of inputs’ and outputs’ delivery dates, which does not leave enough time to work on the input information in between; as opposed to accidental delays resulting from an unexpected outcome of a series of tests on a prototype for example. Even if the product and the information quality are perfect, a structural delaying link will delay the output no matter what because the lack of time is built in.
Once such major process reengineering can be proposed, results should be presented to the division management and even higher, to gain support on implementing the approach and on updating FPDS accordingly. Gaining support for major change will not be easy, that is why a lot of work needs to be done by then.

The new concept of dynamic management of organizations presented in section 5.6 could be implemented if it seems to make any sense to engineers and managers. It would only apply to large enough processes that encompass a true evolution of communication patterns over time. It could be tested on a pilot program with special scrutiny on the efficiency of communication across functions. Also, some more elaborate weighting of the different types of information exchanges could be studied to reflect their relative importance (deliverable contribution information exchanges are arguably more intense than feedbacks and updates from and to recipients after delivery, and even more than simple transmission of the first version of the deliverable to recipients). Instead of simply adding all information flows to create aggregate organizations DSMs, the feedback and update flows and the transmission to recipients could for example be indexed to respectively 0.5 and 0.3. Finally, since FPDS phases were criticized for not aligning correctly with the commodity process, they might not be the right way to slice the process to propose a dynamic management of organizations across phases. More on this issue is explained in chapter 6, section 6.4.

One can also imagine an easier management of the database through an intranet website containing the deliverable DSM and all attached data that would serve as a reference to any design practitioner. Procedures for tracking changes could be established, like ones that Ford already implements through ‘e-tracker’ for the managements of open issues and assignments: anyone willing to change a point of the data could post a request and all impacted teams would have to approve before the data changes. Practically, to avoid the bureaucracy of such procedures, the team willing to update the data could first meet the other teams holding stake in the change (found thanks to the deliverable DSM database), would seek approval and the electronic approval procedure would just be a validation. Many other ways to handle shared knowledge exist and can be invented...

Finally, even further away, a Knowledge Based Engineering (KBE) project could be based on the deliverable DSM interactions. Behind appropriate links between deliverables could be added models that would process the information of the input and give a prediction, an estimate of the output. The recipient team could this way know very soon if the input creates trouble, discovery of issues would be faster and communication would be key to resolve upfront potential issues coming. Of course, implementation would require a great amount of additional work to build the background models and make them credible to the recipient as well as to the input issuer. An important research initiative of the Center for Innovation in Product Development at MIT, called DOME for Distributed Object-base Modeling Environment and led by Dr. David Wallace (see website references in Chapter 7 section 7.2), works to implement such a KBE project based on a DSM.
5.8 Chapter summary and conclusions

There is a long way to go until this whole process improvement project yields full results. All the information presented in this chapter is actually the first step of a broader endeavor. It is however a very fecund first step that included the following accomplishments.

- An end-to-end cross-functional list of process deliverables was agreed on through a collective give and get exercise. Apart from establishing the list of 596 deliverables, the exercise also enabled to quantify the cross-functional shared knowledge on the role of each team with respect to others in the process, which was very poor up to then. It finally helped understand how each team works, in a rather isolated way or in constant collaboration and support with other teams, depending on the requirements of its functions.

- To fully understand the complexity of the process, a new step of data gathering focused on the input / output connections between deliverables, in order to analyze the drivers of process timing and information quality. The data was formatted into a large deliverable Design Structure Matrix (DSM) which was used to fully analyze many aspects of the process. Structural delays and inconsistencies of timing between teams were flagged, deliverables whose quality was criticized by their customers were identified, deliverables with no predecessors and / or which are not used as inputs for any other deliverable were also identified. The apparition of all these phenomena over the course of the process was also studied in order to determine if the early phase of the process was particularly worse than others. A conclusion is still difficult to draw on this question, there are definitely important timing and above all quality issues in the beginning of the process, but their extent is not as compelling as expected. The idea that issues at the beginning of a process are normal and actually healthy is put forward, although the history of the program clearly shows that in this case there were much more issues than “necessary” or contingent.

- From the deliverable data was aggregated information to study communication between functional teams. Enough data was available to put forward the idea of dynamic management of organizations based on the evolution or Organizations DSM over time, as opposed to the static team clusters over the entire process that Organizations DSM usually provide.

Overall, from a larger point of view, a single data gathering approach through the deliverable DSM enabled:

- Detailed process mapping and understanding of its complexity
- A new project management perspective based on the reality of work (instead of milestones whose relevance is often questionable), providing natural metrics for project progress, that is the completion of deliverables over time
- A dynamic and appropriate management of organizations over time
- Capturing key process-level knowledge, sharing, learning and aligning the visions of all functional teams on the process.

The thesis goal was therefore met on the case study presented in this chapter.
To conclude this chapter in a similar way to the PSA Peugeot-Citroën case in chapter 4, the author was again in the middle of an interesting situation of knowledge capture. This time a framework, the Design Structure Matrix, that scales to the complexity of the process studied, was provided to record process-level knowledge (as opposed to the OTTs in the PSA case study). Apart from very rare negative reactions to the task force work requirements by two subprocess owners (see section 5.6.1 for details on the cultural reasons of these reactions), the author felt again great motivation to capture knowledge for the vast majority of subprocess owners. They all spent hours working on recording links between deliverables in their surveys, reported me several data inconsistencies and all in all took the task force project very seriously, in spite of extremely busy schedules with their regular workload. It was a pleasure to work with all of them and to be taken seriously, although being only a new intern freshly arrived from graduate school.
Part IV: Conclusions and Future Research
6 CONCLUSIONS AND RECOMMENDATIONS

This chapter briefly reviews the work described in this thesis, draws the main conclusions, presents recommendations on the conduction of process improvement and/or DSM projects and on the practice of Product Development in general before describing various opportunities of future research.

6.1 Conclusions

This thesis yielded a succession of conclusions that are worth summarizing here.

- Companies can have very well documented processes that were very carefully thought through and still find them widely frustrating, inefficient and misleading. It also appears that companies can think that they have a good process when they actually only have a schedule.

- Participants of large-scale product development processes organized primarily functionally have little knowledge on the way their team relates to others and on the end-to-end process. This phenomenon is not totally new, but chapter 5 evaluated this knowledge numerically with a strikingly low result.

- Process Mapping, Organizations Management, Project Management and Knowledge Management can all be performed efficiently simultaneously using a deliverable DSM approach, even for extremely large and complex processes.
  
  - The deliverable DSM building process identifies the list of deliverables that need to be accomplished over time for the process to be completed. The DSM identifies input and output relationships between deliverables, laying out the information flow dynamics over time. Timing disconnects (and opportunities to shrink the schedule) as well as information quality issues are identified and flagged, ready for improvement measures.

  - When sufficient data is gathered, organizations DSMs can enable a dynamic management of organizations based on team interactions over time, rather than a static picture of optimal organization that organizations DSM usually provide for a whole process. The organizational structure can therefore be recommended to evolve over time to reflect progress of the process in different phases rather than being fixed throughout the process.
o Project Management becomes tremendously more powerful with a deliverable DSM map of the process: progress is easily traceable through the list of deliverables, the process is smoother since information exchange requirements are known to all and anticipated, consequences of potential delays in one or several deliverables are immediately computable thanks to lead time data, the critical path is easily identifiable, and project duration prediction and resource allocation are made easier by the great amount of data.

o Knowledge Management is done automatically like never before while performing the three preceding tasks: end-to-end process knowledge is recorded on a single fairly easy to understand document that can be used for training of new employees / new participants to the process. Knowledge hubs are identified in the network of teams, enabling targeted Knowledge Management policies to yield better efficiency in knowledge documentation and transmission. While doing all this, participants learn more on their process and work efficiency is improved.

- There is room for improvement in the DSM building process itself: starting off with an initial collective meeting to agree on the list of elements to put in row and columns tremendously reduces the number of iterations between the different DSM building tasks. Also, electronic decentralized DSM building can replace effectively the traditional sequence of interviews for large projects under certain conditions. More detail is given on this point in the next section.

- This work has also proven that DSMs can handle process timing, actually the primary thing they handle in Chapter 5, by building a deliverable DSM with a delivery date attached to each deliverable and a time lag (lead time) between inflow and outflow recorded in the interaction information.

- Another underlying conclusion of this work is that no matter how flawed a product development process seems to be (like in the case study of chapter 5 at Ford Motor Company), reasonably good products, in any case with many less apparent defects than their PD process, can still be designed, manufactured and sold on the market. There seems to be a human force (motivation?, management pressure?, both?) that drives an unclear, messy, frustrating and tension-building process to work anyways, indeed in many cases with significant delays and excessive expenses but with a successful product in the end. Such delays and cost are what this work aims at: in our highly competitive world, companies cannot afford anymore to maintain inefficient rough processes. This thesis showed on a particular case how to improve the efficiency and reliability of PD processes, improving cost and time-to-market, therefore enabling and sustaining successful future of the companies at which they take place.

6.2 Recommendations on the conduction of process / DSM projects

Through two years of DSM practice, especially during the two case studies at PSA Peugeot-Citroën and Ford Motor Company, the author gained some experience that the reader might find valuable to benefit from.
Conclusions and Recommendations

6.2.1 DSM building process

In the PSA Peugeot-Citroën case study (Chapter 4), the DSM building process presented in Chapter 3 was implemented literally. In the Ford Motor Company case study (Chapter 5), the process was modified because the author joined a project that had already started with the give and get offsite meeting and because a process relying solely on interviews was simply not practicable. This rather fortunate change in circumstances led the author to reflect on the way to build DSMs more efficiently. Since the DSM building process is very iterative itself, with strong interactions between its different tasks (for example interviews of two different engineer influence each other as one will challenge the interactions claimed by the other), it made sense to analyze the DSM building process using... DSMs! This is sort of a fractal approach, studying PD processes using DSMs, then studying the DSM building process for studying PD processes using DSMs and so on. The author will stop at stage two of this mise en abîme.

Figures 6.2.1.1 presents the task DSM summarizing the DSM building process presented in Chapter 3 and used in the PSA Peugeot-Citroën case study. Figure 6.2.1.2 shows the improvement a collective consensus at the beginning of the process provides.

![Diagram of DSM building process as presented in Chapter 3](image1)

![Diagram of DSM building process with initial collective meeting (Chapter 5)](image2)

Both processes consist in the same tasks, but figure 6.2.1.2 shows how much the iterations are reduced if the system boundaries and the list of system elements are consensed before interviews about element interactions start. That is what happened in the Ford case study as the initial «give and get» offsite meeting established a shared vision of what deliverables compose the commodity design process. Once this list of deliverables was agreed on, it was much easier to study their interactions later on as very few people came back to the author to add a new
deliverable or remove one. This is very different from the regular DSM building process of figure 6.2.1.1 implemented in the PSA Peugeot-Citroën, where the list of design parameters was constantly adjusted as discussions on the interactions were making progress with the engineers.

The DSM building process improvement actually went further in the Ford case study, as the interactions between interviews (electronic surveys in this case) were also all suppressed. Indeed, in the case of mostly disjoined responsibilities in a process (e.g. functional teams), DSMs can be built by "slicing" the data into chunks specifically relevant to one team and assembling all chunks together in the end, with little if any conflict in the data. The DSM building process used for the Ford case study of Chapter 5 is therefore represented in figure 6.2.1.3.

![Figure 6.2.1.3: DSM building process effectively used in the Ford case study (Chapter 5) in a situation where disjoined responsibilities suppress interactions between interviews](image)

This representation of the process used at Ford is a little bit idealized because, as explained in Chapter 5, not all tasks were perfectly prepared, and some rework with feedback was necessary, for example the missing data from the collective offsite meeting that needed to be gathered through a series of individual interviews. However, the evolution from figure 6.2.1.1 to 6.2.1.3 reflects the improvements that were achieved from the PSA case study to the Ford case study.

Electronic surveys are a necessary and feasible alternative to individual interviews in large projects with mostly disjoined responsibilities. Taking advantage of the waterfall process showed in figure 6.2.1.3, each team representative can get a customized survey dealing specifically with the rows of the DSM that are relevant to him. The surveys are in the format of a spreadsheet including an easily programmed interface that helps the user find the cell he is looking for. They can all be generated automatically coding a simple program. This way, time is spared greatly.

Without these improvements in the DSM building process and the use of electronic surveys, the large deliverable DSM constructed in Chapter 5 would not have been feasible in a summer.

6.2.2 Who leads the DSM project

It appeared clearly to the author that results of the process improvement project of Chapter 5 would have been even more important if his expertise on the DSM methodology had been combined to extensive knowledge on the system and process studied. In order to identify process flaws better, it would have tremendously helped to know the process well, that is to have been
Conclusions and Recommendations

part of it long enough to understand the deliverable descriptions. Typically, what the author was able to identify are structural process flaws, like timing disconnects, or specific quality issues that were asked to the subprocess owners. However, the author was not able to fully understand the detailed logic of the process since most of the deliverable descriptions did not mean much to him. The only qualitative finding (i.e. not found through the systematic check for timing disconnects and low quality deliverables) was the misalignment between prototype testing and the design freeze for the next generation of prototype (see Chapter 5 towards the end of section 5.7 for more details).

The author feels that many more qualitative issues would have been identified by an experienced engineer in the process on top of the structural defects. Many qualitative issues were brought up by various participants in the offsite meetings and in the weekly task force meetings, but an experienced employee working full time on the list of deliverables like the author did would probably have found more problems, enabling further improvement of the process.

The point of the author is that process improvement projects would be much more efficient if they were led and accomplished (full time) by experienced engineers trained in the DSM methodology, rather than graduate students specialized in DSMs who slowly learn on the process studied as the project moves on. It takes much more time to understand an extremely complex process like that of Chapter 5 than to learn the DSM methodology or other process mapping methodologies. Since both are required for process improvement in the eyes of the author, companies had better assign a very experienced engineer trained in DSMs to the project.

Unfortunately, the value that these very experienced engineers (e.g. the supervisor of Systems Engineering in Chapter 5) bring into a program is usually too high for companies to draw them aside from the product development action and let them work full time for 6 to 12 months on process improvement, unless radical reengineering is at stake (see chapter 2 for the terminology difference). It is a pity because they are the most relevant people to achieve process improvement. In a way, there is too much focus on short term goals (program progress) to the detriment of long term ones (consistent process). By doing so, work does get done, but with great difficulty all the time because of lousy processes, as opposed to a temporary downturn in PD action in order to yield long term benefits if an experienced engineer is assigned full time to a major process improvement project for many months.

Nevertheless, companies usually prefer to fully exploit the engineers' experience on design and leave process improvement projects to support teams not involved in the process and sometimes graduate students. It is better than nothing, but the author sees this situation as a transition to the self improvement attitude described above. The role of the student is then to show the power of the tools brought in from academia, DSMs in the author's case, to the company so that it understands the value of the methodology, trains its employees on it (sometimes by hiring the student!) and promotes continuous improvement using it. In the end the company should have become autonomous on the methodology brought in by the student and should be able to implement it within each of its processes, without necessarily major help from company process improvement support teams who have little knowledge on the process targeted. The system sustains itself as academia develops new methodologies, sometimes actually by observing companies (knowledge constantly goes both ways), and students still have their role to play, explaining them to companies.
6.2.3 The value of DSMs
The DSM methodology is still in this transition phase: it has increasingly been recognized as a tremendously useful system structure analysis tool, but it still has not been adopted by companies in the training given to all employees. Some companies are gradually evolving on this track, Ford for example, but managers usually still want to know how much value, in monetary terms, there is in the benefits brought about by DSMs before making the costly move of training everybody.

The problem with putting a dollar value on the DSM methodology lies in the intangibility of the improvements and the long time it takes to see them happen. Knowledge (process knowledge, system structure knowledge) is very hard to quantify in monetary terms and the tangible results that it yields take a long time to become visible. Typically, if process knowledge is improved (e.g. mutual awareness of the information exchanges and consistency of timing of subprocesses), improvements on process duration and cost can only be evaluated once the new process is fully implemented, that is, for large projects, several years after the process improvement project took place. Very similar statements can be made on organizations management changes enabled by DSMs. As described in the previous section, since DSM projects are primarily conducted by graduate students, full results of their work only appear once they have graduated, moved to a different life and their project has kind of been forgotten in the company. It is therefore extremely hard to assess the extent to which a DSM project made the company do better. Results presented in Chapter 4 and above all Chapter 5 are undeniable proofs of the utility of the methodology, but quantifying it is not possible at this point.

Some exceptions however exist, in which DSM projects were conducted long enough within companies and yielded remarkable results, demonstrating that the transition described in the previous section is on its way. The example of Antonino P. Zambito at Ford Motor Company is an excellent one. Tony Zambito was a Ford Motor Company car body engineer when he entered in 1998 the System Design and Management program at MIT (part of the Engineering Systems Division). During his stay at MIT, he worked with Dr. Daniel Whitney and Dr. Ali Yassin on DSMs and applied the methodology to car body design at Ford for his Master of Science thesis (Zambito, 2000). The DSM methodology was also improved through his work, leading to a new approach to DSMs, the Do-It-Right-First-Time approach (adding new tasks and / or modifying relationships between elements) as an alternative to resequencing and partitioning (Yassin et al, 2000). As Tony Zambito returned to Ford, he continued to work on the topic of his Master's thesis and obtained support at Ford to work full time on improving the process of car body design using DSMs. This long project is still under way as this thesis is being written and is showing great success in reducing car body design duration and cost as well as improving the PD process robustness (Yassin et al, 2001). Tony Zambito is now leading a company-wide product creation process reengineering effort at Ford using DSMs, which means that Ford has understood the value of DSMs to spread it across the whole company.

This kind of long term involvement is the only way for companies to really measure the value of using DSMs. They can happen through long term partnerships with academic research centers, with Ph.D. students involved over several years in a project, following and tracking its results, or through hiring “DSM alumni” and letting them run large DSM projects for enough time to be able to see results. In this case, academic research still continues to benefit from the alumni work
if the research center keeps in touch with them and the companies, like for Tony Zambito at Ford, to publish interesting findings from long projects.

The CIPD at MIT is for example quite unlikely to track results of the process improvement project presented in Chapter 5 because the author will have graduated and engineers at Ford will probably not have a chance to study in detail how much better the new proposed process is.

6.3 Recommendations on the practice of Product Development

Obviously, all this thesis is about recommending better practices of Product Development. Many of the following points have already been explained with length, so the author will just briefly summarize them.

- Reengineering product development processes, in the sense of Chapter 2, that is radical redesign of a process, appears to be appropriate in some companies where the heritage of traditions is too heavy to allow adaptation to new contexts. Principles of Lean Manufacturing should be applied as much as the transfer is possible (see Chapter 1 for references on this subject) to make PD processes more efficient. In particular, it is crucial to break the functional “fortresses” to promote customer-driven problem-solving attitude in cross-functional teams rather than misunderstanding, rivalry and opposition. By customer-driven problem-solving attitude the author means an approach that makes all participants of a team share the common goal of the team related to the overall satisfaction of the customer, and not simply care about their own contribution to the team work. The usually verified stereotype that designers care mainly about product quality (i.e. design) while manufacturing engineers care mostly about cost is an example of failure to adopt this approach. In this case designers and manufacturers miss the common goal and make it harder to reach because of counterproductive tensions. Customer satisfaction involves both product quality and low cost, designers and manufacturing engineers should work together with only this common goal in mind, not seeing the other as an obstacle to reach one’s own goals.

- In the face of the large risk entailed in the radical change that process reengineering entails, companies might be tempted to go for the incremental improvement path, like the one described in Chapter 5, which is fine. However, companies need to understand that what drives the behaviors of process participants is the organizational structure of the company. Incremental process improvements will have little effect if the process participants still think in terms of their functional fortress. Geographical co-location or cross-functional weekly team meetings are not enough changes to induce a switch from a functional to a problem-solving attitude described in the previous point. For example, it will never happen if the evaluation and the career advance of process participants is still primarily determined by their home functional department rather than by the program they work for. Improvements made in the process like the resolution of timing disconnects and of quality issues presented in Chapter 5 are limited to deceiving results if they are not carried over by a new way of working and of thinking of the work. The end-to-end process approach adopted in Chapter 5 is only a beginning to make process participants understand why their contribution is important and what the common process goals are.
Finally, companies should work on **popularizing process thinking among their engineers.** Too often, the responsibility to think about process, about the way the work is done, is left to a support team outside of the process. Both case studies showed the flaws of such an approach. In Chapter 3, the support team came up with a tool to map processes (OTTs) that the author found inappropriate and that frustrated process participants. In Chapter 5, the support team, like the author, crucially lacked knowledge on the process itself to yield breakthrough process improvements. As described in section 6.2.2, experienced engineers should be at the heart of process thinking, even if that means an underutilization of their skills and experience in the short term: it is a key investment for the future. Especially if the knowledge gained is carefully recorded using appropriate tools, including DSMs, enabling easy access to information by everyone. This way, knowledge management will also gradually become a key corporate value that will help companies face uncertainty and change in their conditions.

### 6.4 Future Research

This thesis triggered several opportunities for future work and research.

- First a very practical issue on the software used for large DSM projects. Chapter 5 showed the limitations of Microsoft\textsuperscript{TM} Excel\textsuperscript{TM} when handling large matrices. Clearly the limitation to 256 columns significantly challenges the ability to conduct studies of large systems. The DSM community might want to switch to different software packages (and convert the codes written for Excel for the partitioning and Montecarlo simulation algorithms), like Corel\textsuperscript{TM} Quattro Pro\textsuperscript{TM}, which do not contain this troublesome limitation. Furthermore, at this point it would be even better to integrate DSMs directly within a database management software because the amount of information recorded in large DSMs is much heavier than just an interaction information. In the case of Chapter 5, for each deliverable in the deliverable DSM, there was a deliverable ID, a set of contributors, a set of recipients and a verbal description. The capabilities of database software for sorting and linking elements go far beyond that of spreadsheet software. It would make sense to code a generic program in a database management software that would enable representing DSMs, generating Gantt charts and do the analysis the MS Excel DSM add-ins do. It would avoid painful data conversions and allow to work directly on the database when building the DS.$M$. Such an integrated piece of software could also be the basis for Knowledge Based Engineering (KBE) applications (see chapter 5 section 5.7)

- One possible extension of the analysis performed in Chapter 5 would be to build a task DSM from the deliverable DSM by regrouping deliverables that are essentially conveying information that results from the same task. In that case the feedback would appear and clustering the design process of the commodity studied at Ford would be possible. Comparison with existing task DSMs on the design of same commodity would show how much was learnt about the system through the detailed deliverable analysis, apart from timing and information quality issues. Substantial knowledge on each and every deliverable is however required to achieve the transformation from deliverable DSM to task DSM, which makes it very hard to do.
Conclusions and Recommendations

- The notion of dynamic organizations management, described in Chapter 5 section 5.6, needs more "science" to it. Although it is already clear that different phases of the process show different organizational clusters, since the phases defined by FPDS were criticized for being imposed by the vehicle level and inconsistent with the commodity process, it would be interesting to study rigorously the evolution of the organization DSM over time to try to define natural phases in the process. There is great potential in using the statistical models developed by Dr. Manuel Sosa to analyze and compare DSMs (Sosa et al, 2002) toward that goal. Computing over time t correlation factors between the probabilistic distributions of the organization DSMs from time t₀ to time t₁ and from time t₀ to time t₁+t might enable to identify the end of a "natural phase" of the organization dynamics when the correlation factor drops below a certain threshold. This way these natural phases could be compared with the FPDS phases (between two milestones) and with the proposed reviews based on deliverables ("justified decisions") as defined in Chapter 5 section 5.7.

- There is also great potential in building a bridge between DSMs and the theory of social networks. The theory of social networks is widely used by social scientists to analyze the structure of networks in general through quantitative metrics like the connectivity of a node (the number of other nodes it is connected to), the connectivity of a network (i.e. the minimum number of links traveled through to reach all nodes from all other nodes), the accessibility of a node (i.e. the minimum number of links traveled through to reach all or certain nodes from a specific node), the beta index (ratio of the number of links over the number of nodes in a network), the diameter (number of links in the shortest path between the furthest pair of nodes), etc. All are used to determine the topology of networks, compare them and explain observed properties like stability or searchability. Popular applications touch food webs (Neute et al, 2002), protein / metabolic networks (Ravesz et al, 2002; Maslov and Sneppen, 2002) and human / social networks (Duncan, Sheridan Dodds, Newman, 2002). There is also significant research done on software applications to help represent networks graphically (McMahon, Miller and Drake, 2001). No application of this theory to engineering can however be found, although DSMs are simply a different representation of networks emphasizing their structure, and product development is a social network process in which people are the nodes and information is exchanged along the links. There is therefore no reason for this set of tools not to be applied to Product Development, and vice versa for the DSM with food webs, protein networks and human networks outside of the engineering world. It would be interesting to understand how to compute network structure metrics from DSMs in order to see what extra information can be extracted from them using the social networks theory approach. Such computation could be included in the DSM macros and plug-ins for Excel, enabling a systematic social networks analysis of previously built DSMs. That way the social networks aspect of all previous studies using DSMs could be easily revealed, and it would be very interesting to see how much more can be learnt on these systems from the metrics and their interpretation. This way the true value of combining DSMs with social networks analysis metrics could be identified, maybe with the help of social networks analysts. It may be that there is not much new information to be learnt, in which case it would mean that the DSM analysis is providing equivalent conclusions to the social networks one. It may also be that some valuable information was hidden in the DSMs and is revealed by the calculation of the social networks metrics, in which case the value of combining both fields would be proven. A symmetrical review could also be performed by some social networks analysts who might as well find new
interesting conclusions by applying the DSM clustering approach to the networks they study. Also, it would be interesting to benefit from the visualization software used by social networks analysts and try to apply it to representing a network described by a DSM and see again how useful it is.

- Finally, the author feels that there is still a lot to be learnt from the meaning and the identification of the roots of the Whitney ratio (i.e. the average number of marks per row of a DSM). As described in Chapter 3 section 3.4, there seems to be a fairly consistent average of around 6.2 marks per row over a wide range of DSM project sizes. However, in both case studies, although using interviews in one case and electronic surveys in the other, the author found averages significantly lower than 6.2, namely 3.3 for the FEAD system design parameter DSM (4.2 if restricting the DSM to the parameters of the system only) and 3.4 for the Ford Motor Company commodity design process deliverable DSM. These numbers are different enough from the average observed, and similar enough between both cases studies to raise a set of questions:
  - How much does the Whitney ratio depend on the person who conducted the study? Can differences be identified between sets of DSMs built by the same author? Would the way the DSM methodology is presented or the way interviews are set up influence the Whitney ratio? One could argue that interviews allow more information to be captured since the interviewer raises issues and questions as the interviewee fills the matrix, as opposed to when the interviewee is filling up a survey on his or her own. But both cases studies of this thesis yielded the same average number of marks per row although they did not use the same data collection approach. More than two case studies would be needed to answer that question.
  - Is there any correlation pattern with respect to the Whitney ratio between DSMs of the same type (task, design parameter, organization, physical interaction)? Can DSMs of a different type be really compared easily or should the Whitney ratios be compared within each type of DSM? This question particularly applies to the deliverable DSM since it records interactions between documents, and documents are known to contain less knowledge that the system requires.

Also, if a possible explanation of the Whitney ratio is a universal human mental structure that leads humans, at least in the engineering world, to design systems with no more than an average of 6 connections with other parameters (simple systems with much less interactions are likely not to be studied with DSMs), some experiments could be designed with Brain and Cognitive Sciences researchers to test that hypothesis.
Conclusions and Recommendations
7 BIBLIOGRAPHY

7.1 References


Dong, Q. (2002), "Predicting and Managing System Interactions at Early Phase of the Product Development Process", Ph.D. Thesis, Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA


Bibliography


7.2 Sponsors’ and support websites

The National Science Foundation
http://www.nsf.gov

MIT Center for Innovation in Product Development
http://web.mit.edu/cipd
   The IFM page (including on DSMs): http://web.mit.edu/cipd/research/ifm.htm
   The DOME page: http://web.mit.edu/cipd/research/dome.htm
   The DSM website: http://web.mit.edu/dsm/

Dr Daniel E. Whitney’s personal web page
http://web.mit.edu/cipd/www/Whitney/papers.html

The Franco-American Commission for Educational Exchange
http://www.fulbright-france.org

The Fulbright Program
http://www.fulbrightweb.org

Fondation de l’Ecole Polytechnique
http://www.fondationx.org/

Renault Group
http://www.renault.com
Appendices
Appendix A: Abstract of PSA’s organization chart
(only groups involved in the case study of Chapter 4 are represented)
## Appendix B: Give & Get data table

<table>
<thead>
<tr>
<th>Organization</th>
<th>Total Cards</th>
<th>Redundant cards</th>
<th>Deliverable-like cards (=total - redundant - matches)</th>
<th>Unmatched Gives</th>
<th>% of deliverable-like cards</th>
<th>Unmatched Gets</th>
<th>% of deliverable-like cards</th>
<th>Matches</th>
<th>% of deliverable-like cards</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ASSUMPTIONS &amp; TARGETS</strong></td>
<td>46</td>
<td>14</td>
<td>26</td>
<td>14</td>
<td>54%</td>
<td>6</td>
<td>23%</td>
<td>6</td>
<td>23%</td>
</tr>
<tr>
<td>Assumptions</td>
<td>34</td>
<td>14</td>
<td>17</td>
<td>13</td>
<td>76%</td>
<td>1</td>
<td>6%</td>
<td>3</td>
<td>16%</td>
</tr>
<tr>
<td>Targets</td>
<td>12</td>
<td>0</td>
<td>9</td>
<td>1</td>
<td>11%</td>
<td>5</td>
<td>56%</td>
<td>3</td>
<td>33%</td>
</tr>
<tr>
<td><strong>ANALYTICAL TEAMS</strong></td>
<td>55</td>
<td>2</td>
<td>51</td>
<td>36</td>
<td>71%</td>
<td>13</td>
<td>25%</td>
<td>2</td>
<td>4%</td>
</tr>
<tr>
<td>CAE Durability</td>
<td>17</td>
<td>0</td>
<td>17</td>
<td>13</td>
<td>78%</td>
<td>4</td>
<td>24%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>CAE team 2</td>
<td>7</td>
<td>0</td>
<td>7</td>
<td>7</td>
<td>100%</td>
<td>0</td>
<td>0%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>CAE NVH (Noise Vibration Harshness)</td>
<td>11</td>
<td>0</td>
<td>11</td>
<td>10</td>
<td>91%</td>
<td>1</td>
<td>9%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>CAE team 4</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>3</td>
<td>60%</td>
<td>2</td>
<td>40%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>CAE team 5</td>
<td>15</td>
<td>2</td>
<td>11</td>
<td>3</td>
<td>27%</td>
<td>6</td>
<td>55%</td>
<td>2</td>
<td>18%</td>
</tr>
<tr>
<td><strong>DESIGN</strong></td>
<td>135</td>
<td>2</td>
<td>129</td>
<td>33</td>
<td>26%</td>
<td>92</td>
<td>71%</td>
<td>4</td>
<td>3%</td>
</tr>
<tr>
<td>Systems Engineering</td>
<td>56</td>
<td>0</td>
<td>55</td>
<td>20</td>
<td>36%</td>
<td>32</td>
<td>56%</td>
<td>3</td>
<td>5%</td>
</tr>
<tr>
<td>CAD Design</td>
<td>23</td>
<td>0</td>
<td>23</td>
<td>2</td>
<td>9%</td>
<td>21</td>
<td>91%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>Component Engineering</td>
<td>54</td>
<td>2</td>
<td>51</td>
<td>11</td>
<td>22%</td>
<td>39</td>
<td>76%</td>
<td>1</td>
<td>2%</td>
</tr>
<tr>
<td><strong>TESTING</strong></td>
<td>52</td>
<td>3</td>
<td>44</td>
<td>27</td>
<td>61%</td>
<td>12</td>
<td>27%</td>
<td>5</td>
<td>11%</td>
</tr>
<tr>
<td>Testing team 1</td>
<td>33</td>
<td>3</td>
<td>27</td>
<td>14</td>
<td>52%</td>
<td>10</td>
<td>37%</td>
<td>3</td>
<td>11%</td>
</tr>
<tr>
<td>Durability Testing</td>
<td>19</td>
<td>0</td>
<td>17</td>
<td>13</td>
<td>76%</td>
<td>2</td>
<td>12%</td>
<td>2</td>
<td>12%</td>
</tr>
<tr>
<td>Manufacturing Engineering</td>
<td>18</td>
<td>1</td>
<td>16</td>
<td>10</td>
<td>63%</td>
<td>5</td>
<td>31%</td>
<td>1</td>
<td>6%</td>
</tr>
<tr>
<td>Planning, Higher level Management</td>
<td>52</td>
<td>2</td>
<td>49</td>
<td>12</td>
<td>24%</td>
<td>36</td>
<td>73%</td>
<td>1</td>
<td>2%</td>
</tr>
<tr>
<td>Procure &amp; Build, Logistics</td>
<td>18</td>
<td>0</td>
<td>18</td>
<td>15</td>
<td>83%</td>
<td>3</td>
<td>17%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>Purchasing - Sourcing</td>
<td>18</td>
<td>2</td>
<td>16</td>
<td>8</td>
<td>50%</td>
<td>8</td>
<td>50%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td><strong>QUALITY &amp; STA</strong></td>
<td>20</td>
<td>0</td>
<td>19</td>
<td>18</td>
<td>95%</td>
<td>0</td>
<td>0%</td>
<td>1</td>
<td>5%</td>
</tr>
<tr>
<td>Commodity Division Quality</td>
<td>12</td>
<td>0</td>
<td>12</td>
<td>12</td>
<td>100%</td>
<td>0</td>
<td>0%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>Supplier Technical Assistance</td>
<td>8</td>
<td>0</td>
<td>7</td>
<td>6</td>
<td>86%</td>
<td>0</td>
<td>0%</td>
<td>1</td>
<td>14%</td>
</tr>
<tr>
<td><strong>PROTOTYPING SHOP</strong></td>
<td>16</td>
<td>4</td>
<td>8</td>
<td>0</td>
<td>0%</td>
<td>4</td>
<td>50%</td>
<td>4</td>
<td>50%</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td>430</td>
<td>30</td>
<td>376</td>
<td>173</td>
<td>46%</td>
<td>179</td>
<td>48%</td>
<td>24</td>
<td>6%</td>
</tr>
</tbody>
</table>

### REMOVED BECAUSE PARTIAL INFORMATION AVAILABLE ONLY

<table>
<thead>
<tr>
<th>Organization</th>
<th>Total</th>
<th>Redundant</th>
<th>Deliverable-like cards (=total - redundant - matches)</th>
<th>Unmatched</th>
<th>% of deliverable-like cards</th>
<th>Unmatched</th>
<th>% of deliverable-like cards</th>
<th>Matches</th>
<th>% of deliverable-like cards</th>
</tr>
</thead>
<tbody>
<tr>
<td>Testing team 2</td>
<td>21</td>
<td>0</td>
<td>21</td>
<td>21</td>
<td>100%</td>
<td>0</td>
<td>0%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>Advanced Engineering, Ford Research Lab</td>
<td>8</td>
<td>0</td>
<td>8</td>
<td>6</td>
<td>17%</td>
<td>5</td>
<td>83%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>Vehicle interface team</td>
<td>3</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0%</td>
<td>3</td>
<td>100%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>CAE Team 6</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0%</td>
<td>1</td>
<td>100%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>Finance</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0%</td>
<td>1</td>
<td>100%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>Ford Land</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0%</td>
<td>1</td>
<td>100%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>Commodity Division Software team</td>
<td>7</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>0%</td>
<td>7</td>
<td>100%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>Commodity Testing Facility</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0%</td>
<td>2</td>
<td>100%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>Suppliers (parts &amp; machines)</td>
<td>12</td>
<td>0</td>
<td>12</td>
<td>0</td>
<td>0%</td>
<td>12</td>
<td>100%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>Vehicle Operations</td>
<td>11</td>
<td>1</td>
<td>10</td>
<td>0</td>
<td>0%</td>
<td>10</td>
<td>100%</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td><strong>TOTAL REMOVED</strong></td>
<td>65</td>
<td>1</td>
<td>64</td>
<td>22</td>
<td>34%</td>
<td>42</td>
<td>66%</td>
<td>0</td>
<td>0%</td>
</tr>
</tbody>
</table>

**Comments:**
- Assumptions and targets: match counting technique favors assumptions and targets since their deliverables are distributed to everyone, only one recipient request and it is considered a match
- Planning: definition of the boundaries of the team not understood by all, ambiguity + similar bias for matches
- Performance mapping: all recipients of gives not represented at the meeting, no possibility for match
- All other removed: not present at offsite, only get cards put by teams represented
## Appendix C: Deliverables data table

<table>
<thead>
<tr>
<th>Organization</th>
<th>Owned</th>
<th>% of total of the row</th>
<th>Contributed to</th>
<th>% of total of the row</th>
<th>Received</th>
<th>% of total of the row</th>
<th>Total (excl. redundants)</th>
<th>% of total # of deliverables</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASSUMPTIONS &amp; TARGETS</td>
<td>34</td>
<td>29%</td>
<td>20</td>
<td>17%</td>
<td>77</td>
<td>66%</td>
<td>116</td>
<td>19%</td>
</tr>
<tr>
<td>Assumptions</td>
<td>24</td>
<td>33%</td>
<td>16</td>
<td>22%</td>
<td>46</td>
<td>63%</td>
<td>73</td>
<td>12%</td>
</tr>
<tr>
<td>Targets</td>
<td>10</td>
<td>23%</td>
<td>4</td>
<td>9%</td>
<td>31</td>
<td>72%</td>
<td>43</td>
<td>7%</td>
</tr>
<tr>
<td>ANALYTICAL TEAMS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAE Durability</td>
<td>21</td>
<td>19%</td>
<td>7</td>
<td>8%</td>
<td>60</td>
<td>71%</td>
<td>85</td>
<td>14%</td>
</tr>
<tr>
<td>CAE team 2</td>
<td>11</td>
<td>17%</td>
<td>5</td>
<td>8%</td>
<td>49</td>
<td>78%</td>
<td>63</td>
<td>11%</td>
</tr>
<tr>
<td>CAE NVH (Noise Vibration Harshness)</td>
<td>13</td>
<td>18%</td>
<td>5</td>
<td>7%</td>
<td>57</td>
<td>79%</td>
<td>72</td>
<td>12%</td>
</tr>
<tr>
<td>CAE team 4</td>
<td>7</td>
<td>11%</td>
<td>6</td>
<td>10%</td>
<td>52</td>
<td>84%</td>
<td>62</td>
<td>10%</td>
</tr>
<tr>
<td>CAE team 5</td>
<td>15</td>
<td>21%</td>
<td>18</td>
<td>25%</td>
<td>42</td>
<td>59%</td>
<td>71</td>
<td>12%</td>
</tr>
<tr>
<td>DESIGN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Systems Engineering</td>
<td>81</td>
<td>21%</td>
<td>76</td>
<td>20%</td>
<td>283</td>
<td>70%</td>
<td>378</td>
<td>63%</td>
</tr>
<tr>
<td>CAD Design</td>
<td>25</td>
<td>30%</td>
<td>22</td>
<td>27%</td>
<td>42</td>
<td>51%</td>
<td>83</td>
<td>14%</td>
</tr>
<tr>
<td>Component Engineering</td>
<td>69</td>
<td>26%</td>
<td>88</td>
<td>33%</td>
<td>152</td>
<td>57%</td>
<td>267</td>
<td>45%</td>
</tr>
<tr>
<td>TESTING</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Testing team 1</td>
<td>36</td>
<td>43%</td>
<td>8</td>
<td>10%</td>
<td>41</td>
<td>49%</td>
<td>84</td>
<td>14%</td>
</tr>
<tr>
<td>Durability Testing</td>
<td>16</td>
<td>31%</td>
<td>5</td>
<td>10%</td>
<td>33</td>
<td>63%</td>
<td>52</td>
<td>9%</td>
</tr>
<tr>
<td>Testing team 2</td>
<td>30</td>
<td>45%</td>
<td>0</td>
<td>0%</td>
<td>36</td>
<td>55%</td>
<td>66</td>
<td>11%</td>
</tr>
<tr>
<td>Manufacturing Engineering</td>
<td>51</td>
<td>44%</td>
<td>8</td>
<td>7%</td>
<td>73</td>
<td>63%</td>
<td>116</td>
<td>19%</td>
</tr>
<tr>
<td>Planning, Higher level Management</td>
<td>62</td>
<td>48%</td>
<td>3</td>
<td>2%</td>
<td>71</td>
<td>55%</td>
<td>129</td>
<td>22%</td>
</tr>
<tr>
<td>Procure &amp; Build, Logistics</td>
<td>18</td>
<td>22%</td>
<td>13</td>
<td>16%</td>
<td>56</td>
<td>67%</td>
<td>83</td>
<td>14%</td>
</tr>
<tr>
<td>Purchasing - Sourcing</td>
<td>20</td>
<td>22%</td>
<td>13</td>
<td>14%</td>
<td>60</td>
<td>67%</td>
<td>90</td>
<td>15%</td>
</tr>
<tr>
<td>QUALITY &amp; STA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quality</td>
<td>25</td>
<td>20%</td>
<td>24</td>
<td>19%</td>
<td>76</td>
<td>61%</td>
<td>125</td>
<td>21%</td>
</tr>
<tr>
<td>Supplier Technical Assistance</td>
<td>12</td>
<td>28%</td>
<td>2</td>
<td>5%</td>
<td>29</td>
<td>67%</td>
<td>43</td>
<td>7%</td>
</tr>
<tr>
<td>PROTOTYPING SHOP</td>
<td>8</td>
<td>21%</td>
<td>1</td>
<td>3%</td>
<td>30</td>
<td>77%</td>
<td>39</td>
<td>7%</td>
</tr>
<tr>
<td>TOTAL</td>
<td>542</td>
<td>27%</td>
<td>322</td>
<td>16%</td>
<td>1270</td>
<td>64%</td>
<td>1981</td>
<td>332%*</td>
</tr>
</tbody>
</table>

### Notes:

- **NOT PRESENT AT THE INITIAL OFFSITE MEETING:** DELIVERABLES CLAIMED BY TEAMS PRESENT ONLY
- **Advanced Engineering, Ford Research Lab:** 7 44% 9 56% 1 6% 16 3%
- **Vehicle subsystem interfacing with commodity:** 0 0% 1 50% 1 50% 2 0%
- **Vehicle Interface team:** 4 16% 1 4% 0 80% 25 4%
- **CAE Team 6 (vehicle level):** 1 4% 0 0% 0 96% 2 5%
- **Finance:** 1 4% 7 30% 0 87% 2 4%
- **Ford Land:** 2 100% 1 0% 0 0% 1 0%
- **Ford Customer Service Division:** 0 0% 1 100% 1 0% 1 0%
- **Commodity Division Software Team:** 7 100% 0 0% 0 0% 1 0%
- **Commodity Testing Facility:** 2 11% 0 17% 89% 19 3%
- **Suppliers (parts & machines):** 20 24% 15 18% 51 60% 85 14%
- **Vehicle Operations:** 10 38% 4 15% 12 40% 26 4%

### Total Claimed by Teams Present

|                      | 54 23% | 38 16% | 149 64% | 234 39% |

**Comments:**

Redundancies (e.g., when the lead is also contributor or recipient) are removed from the total statistics (last column) of each row.

*However, consolidated statistics, taking into account several teams/rows, inevitably incorporate redundancies in the number of deliverables, one team being the lead and the other being a recipient will count as 2 deliverables although it is only one. Percentages in the 'total column can therefore exceed 100%, but they have to be compared with the overall percentage of 332% which takes all redundancies into account. Unlike table 5.4.3, all other percentages indicate the distribution of activity (lead, contribution, reception) within a row.
Appendix D: Deliverable DSM survey written instructions

Please print (color) and read carefully before starting working on the survey.

I. [Task force name] background information

[Task Force name] is a team formed in January 2002 to assess and improve the [commodity] design & development process by resolving the timing and content disconnects throughout the various organizations involved. It is championed by Thomas McCarthy and supervised by Chris Minger, the [Prog] program being the support of the project. A "give and get" deliverable approach led to an offsite meeting on June 4, 2002 where all process owners put their give and get cards (deliverables) on a wall. Some disconnects and issues were identified, some of them resolved directly, the rest remaining open.

The team has been working hard since then to process the data collected and resolve the remaining open issues. Many of them were resolved through small-scale meetings and the proposed modifications were taken into account in the database. For example, the whole pre-KO process and the assumptions and targets work stream were largely modified in order to address issues identified at the offsite. We are now moving toward the next step: identifying inputs for each deliverable, in order to fully understand what drives delays, disconnects and rework. The data collection is done through the present survey. Results of this project are expected to lead to major process adjustments, and not for [Commodity] design & development only. Additionally, it is to serve as best practice to develop the [Commodity division] PDQOS (Product Development Quality Operating System).

The way deliverables were modeled includes:

- A unique deliverable ID in the format XXX99a, where XXX is the source organization code, 99 is the delivery time in the "month before job 1" scale (and NOT the time when the source organization starts working on it) and the last letter is the "rank" of the deliverable among others issued by the same source organization at the same date. For processing purposes, a unique number is associated to each deliverable (from 1 to 596), with no particular meaning.
- Additional contributors if the deliverable is prepared by more than one organization, but a unique single lead source was always identified for every deliverable, even if it results from a joint effort like a PAT.
- Identified recipients
- Deliverable Description

The complete work stream for your organization, including the deliverables you produce, contribute to and receive sorted by delivery date, is attached to the survey for your information.
II. Organizations: codes & comments

In order to obtain a useful output for process improvement, organizations involved in the engine design process had to be modeled in a slightly different way from the actual organization chart in place. Explanations follow.

[Some parts removed for confidentiality reasons]

- The [Systems Engineering team] having so many different activities, it made sense to separate some of them into:
  - Design & development activities (DEP)
  - Assumptions tracking and setting (ASM)
  - Targets tracking and setting (TRG).

- [Prototyping Shop team] is the organization that delivers build [commodities], since that is where they are assembled ([Systems Engineering team] - Design (DEP) is responsible for designing the builds, Procure and Build (PPM) for ordering and managing shipment of parts).

- CAE activities are performed by different organizations
  - [Analytical teams & codes]
  - [Component Engineering teams] (DCP) also run CAE models on their own, and this activity was agreed to remain included in the Component Design activity

- No distinction is made between all different components: component design is always coded as a single activity, DCP.

- No distinction is made either between part and machine suppliers, they are all coded as SUP

- Because the focus is on the [commodity] design process, it also made sense to merge higher-level management and decision makers into one code, together with the planning activity (PLN). Henceforth, PLN includes people and organizations as varied as:
  - Program planning
  - The program manager
  - [Commodity Division] Strategy & Business Office
  - [Commodity Division] Governance Board
  - The vehicle teams (only their decision-making activities, operations are coded separately)
  - Government affairs
  - Human Resources

- Vehicle operations (VOP) are clearly distinguished from Vehicle PMTs, included in PLN. They are responsible for vehicle builds.
The resulting organizations list follows.

<table>
<thead>
<tr>
<th>Organization Description</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advanced Engineering (incl. Ford Research Lab)</td>
<td>AEG</td>
</tr>
<tr>
<td>Assumptions (non-design activity of [Systems Engineering])</td>
<td>ASM</td>
</tr>
<tr>
<td>CAD</td>
<td>DCD</td>
</tr>
<tr>
<td>CAE – Team 1</td>
<td>Code1</td>
</tr>
<tr>
<td>CAE – Team 2</td>
<td>Code2</td>
</tr>
<tr>
<td>CAE – Team 3</td>
<td>Code3</td>
</tr>
<tr>
<td>CAE – Team 4</td>
<td>Code4</td>
</tr>
<tr>
<td>CAE – Team 5</td>
<td>Code5</td>
</tr>
<tr>
<td>[Component Engineering] (+ their own CAE)</td>
<td>DEP</td>
</tr>
<tr>
<td>[Prototyping shop]</td>
<td>Code</td>
</tr>
<tr>
<td>Testing – Team 1</td>
<td>Code</td>
</tr>
<tr>
<td>Testing – Team 2</td>
<td>Code</td>
</tr>
<tr>
<td>[Durability Testing]</td>
<td>Code</td>
</tr>
<tr>
<td>[Systems Engineering team] (design activity only)</td>
<td>FIN</td>
</tr>
<tr>
<td>Finance</td>
<td></td>
</tr>
<tr>
<td>Ford Customer Service Division</td>
<td>FCS</td>
</tr>
<tr>
<td>Ford Land</td>
<td>FLD</td>
</tr>
<tr>
<td>CAE – Team 6</td>
<td>Code6</td>
</tr>
<tr>
<td>Manufacturing Engineering</td>
<td>MFG</td>
</tr>
<tr>
<td>[Software Engineering]</td>
<td>Code</td>
</tr>
<tr>
<td>Planning (including all higher-level management)</td>
<td>PLN</td>
</tr>
<tr>
<td>[Commodity Testing Facility]</td>
<td>Code</td>
</tr>
<tr>
<td>[Commodity Division] Quality</td>
<td>QUA</td>
</tr>
<tr>
<td>Procurement &amp; Build, Materials &amp; Logistics</td>
<td>PPM</td>
</tr>
<tr>
<td>[Interface with vehicle]</td>
<td>Code</td>
</tr>
<tr>
<td>Purchasing</td>
<td>PUR</td>
</tr>
<tr>
<td>STA - Supplier Technical Assistance</td>
<td>STA</td>
</tr>
<tr>
<td>Suppliers (parts &amp; machines)</td>
<td>SUP</td>
</tr>
<tr>
<td>Targets</td>
<td>TRG</td>
</tr>
<tr>
<td>Vehicle Operations (ops only, management part of planning)</td>
<td>VOP</td>
</tr>
</tbody>
</table>

### III. **Survey specific directions**

The format chosen for the survey to determine interactions between deliverables (which ones serve as inputs for other ones) is a BIG matrix. The layout below will help you understand it.
[this is the real way the survey was laid out, the one presented in Chapter 5 is the transposed for consistency and clarification purposes]
Sequence of required steps:

You are required to accept to enable macros when you open the survey file.
1. Answer the preliminary questions for each deliverable you produce or contribute to. (first block of rows) Upon completion, hide these rows and these rows only.
2. Read the example columns, when you’ve understood them, hide them and the instructions row as well. From now on you must not hide any more rows.
3. For each deliverable you produce or contribute to, column after column:
   a. Think of which DIRECT inputs you need. By direct, we mean that you don’t want to include inputs of your inputs in your own list of inputs.

In order not to forget inputs, you may also scan the block of inputs you are reported to receive (on top of the inputs list) if there are not too many of them. It might remind you of another input you need.

b. You should now have a clear idea of what the inputs of the deliverable / column you’re working on are, launch the input search engine, select the organization that you think is lead on the input you’re thinking of, (optional) enter when you think this deliverable is ready and click "Find". The engine shows you all the deliverables the organization you have selected produces or contributes to within a range of + or – 4 months around the date you entered. All of them appear if no date was entered. Find the deliverable you are looking for by reading the descriptions, select it and click on "Go to record". It will simply scroll the file down for you until the exact row of your input.
If you don’t find what you’re looking for, you can refine / change your search, and if the input you are looking for is really missing, you can always mention it in a comment typed in your corresponding deliverable row.

c. You now know in which cell to enter your data. Type in three characters indicating:
   • How many WEEKS you need the input in advance of your own delivery date.
   Do not base this information on the difference between reported delivery dates of the
deliverable and the input (which can be negative if there is a large disconnect), but rather on the real amount of time you need to have your input information in advance. To simplify, 1 month = 4 weeks

- The level of trust you have in the quality / accuracy of the input on a Low (L) / Medium (M) / High (H) scale
- The impact the quality of the input will have on the quality of your deliverable (output), on a Low impact (L) / Medium impact (M) / High impact (H).

For example, 6HL indicates that you need the input 6 weeks before you can deliver your output, you highly trust the accuracy of the input but it would have a low impact on the quality of your output anyway.

d. Jump to the right side of the spreadsheet by hitting the "end" button and the right arrow, and answer questions about feedback you would give to the source of the input. If you want to add comments on the input, that is what the last column is made for. Any remark is welcome.

e. Repeat steps b., c. and d. as many times as the number of different inputs you have identified for the deliverable / column considered. When you're done with all the inputs of this deliverable, hide its column (select the column tab, right click and select hide)

4. Repeat step 3 until you have typed the data on your inputs for all your deliverables / columns.

If you also have comments on your own deliverables (on contributors, recipients, the way they are described, the time they are reported to be delivered at or to report a missing input for example), you can also do so in the corresponding row in the rightmost column. In particular, your comments will be helpful for the resolution or even discovery of disconnect / open issues.

If you want to go back to a deliverable you have already worked on and hidden, you have to unhide all hidden columns by selecting the unhiden surrounding ones, right clicking and selecting unhide. Do not forget to hide them all again when you're done modifying the data you entered or entering new data.

Do's and don'ts

- Use full screen view mode (in the view menu) to increase the useable area of your screen
- All the data is to be entered in the survey worksheet. Do NOT modify other cells than those where you are supposed to type your data in. If you have comments, include them in the comment section. All the other sheets should not be modified, the organization codes list is available in the second sheet for your convenience, and all other sheets are background data for the input engine search macro. They are NOT protected (and can't be).
- Save regularly and in different files as you progress through your deliverables. You don't want to have to do this twice if there is a power breakdown or if your file is corrupted.
- Hide columns as you progress through your deliverables, it will avoid you annoying scrolling when you use the input search engine
- Do NOT hide input rows, it might cause the search macro to crash
- Hit the "END" key and the right arrow to jump to the questions & comments section, don't bother scrolling
- Please respect your deadline date, we need to finish collecting the data quickly to start analyzing it shortly.
INSTRUCTIONS SUMMARY
Please print and keep with you while you work on the survey.

Sequence of required steps:

You are required to accept to enable macros when you open the survey file.
1. Answer the preliminary questions for each deliverable you produce or contribute to.
   (first block of rows) Upon completion, hide these rows and these rows only.
2. Read the example columns, when you've understood them, hide them & the instructions row. From now on you must not hide any more rows.
3. For each deliverable you produce or contribute to, column after column:
   a. Think of which DIRECT inputs you need. You may scan the list of inputs you are reported to receive if it is not too long, it might help you remember of inputs.
   b. You should now have a clear idea of what the inputs for the deliverable / column you're working on are, use the input search engine to find one of them
   c. You now know in which cell to enter your data. Type in three characters indicating:
      • How many WEEKS you need the input in advance of your own delivery date.
        (To simplify, 1 month = 4 weeks)
      • The level of trust you have in the quality / accuracy of the input on a Low (L) / Medium (M) / High (H) scale
      • The impact the quality of the input will have on the quality of your deliverable (output), on a Low impact (L) / Medium impact (M) / High impact (H).
   d. Jump to the right side of the spreadsheet, answer questions about feedback you would give to the source of the input, and add comments if necessary.
   e. Repeat steps b., c. and d. as many times as the number of inputs you have identified for the deliverable / column considered. When you're done with all the inputs of this deliverable, hide its column (select the column tab, right click and select hide).
4. Repeat step 3 until you have typed the data on your inputs for all your deliverables / columns. Please also add comments on your own deliverables in the corresponding rows.

Do's and don'ts

• Use full screen view mode (in the view menu) to increase the useable area of your screen
• All the data is to be entered in the survey worksheet. Do NOT modify other cells than those where you are supposed to type your data in. If you have comments, include them in the comment section. All the other sheets should not be modified, the organization codes list is available in the second sheet for your convenience, and all other sheets are background data for the input engine search macro. They are NOT protected (and can't be).
• Save regularly and in different files as you progress through your deliverables. You don't want to have to do this twice if there is a power breakdown or if your file is corrupted.
• Hide columns as you progress through your deliverables, it will avoid you annoying scrolling when you use the input search engine
• Do NOT hide input rows, it might cause the search macro to crash
• Hit the "END" key and the right arrow to jump to the questions & comments section, don't bother scrolling
• Please respect your deadline, we need to finish collecting the data quickly to start analyzing it.
Appendix E : Excerpts of the deliverable DSM

This first page is an example with the legend, real data follows in the next pages.