Open Collaborative System Design

A Strategic Framework with Application to Synthetic Biology

Matthew Robin Silver

B.A. Astrophysics, Art History, Williams College, 2001
S.M. Aerospace Engineering, Massachusetts Institute of Technology, 2005
S.M. Technology and Policy, Massachusetts Institute of Technology, 2005

SUBMITTED TO THE ENGINEERING SYSTEMS DIVISION IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY
AT THE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2010

Author _____
Engineering Systems Division
January 2010

Certified by _____
Edward F. Crawley
Professor of Aeronautics and Astronautics and Engineering Systems
Thesis Supervisor

Certified by _____
Thomas Knight
Senior Research Scientist, Electrical Engineering and Computer Science

Certified by _____
Joel Moses
Institute Professor
Department of Electrical Engineering and Computer Science and Engineering Systems

Certified by _____
Kenneth Oye
Associate Professor of Political Science and Engineering Systems

Certified by _____
Olivier L. de Weck
Associate Professor of Aeronautics and Astronautics and Engineering Systems

Accepted by _____
Nancy Leveson
Professor of Engineering Systems and Aeronautics & Astronautics
Chair, ESD Education Committee

Open Collaborative System Design

A Strategic Framework with Application to Synthetic Biology

by,

Matthew Robin Silver

Submitted to the Engineering Systems Division
on January 8th, 2010 in Partial Fulfillment of the Requirements for the
Degree of Doctor of Philosophy in
Engineering Systems

Abstract

Across technology industries and particularly at the cutting edge of biotechnology a debate is under way about the proper balance between open and closed – between co-developing products with shared information and open standards, versus using more traditional, closed, proprietary processes. Beyond the relative success of open source software to date, it is not clear how and whether open design processes might be applied generally for complex, assembled technologies. This problem takes on special urgency within the domain of synthetic biology, an emerging discipline in which many practitioners advocate opening design and development through platforms such as the registry of standardized biological parts. Biotechnology is IP intensive in part because commercialization is complicated and capital intensive. How might one develop a sustainable open development process in this context?

This thesis addresses these questions from an Engineering Systems perspective. Defining open, collaborative system development (OCSD) specifically as a process in which *sub-systems* are created voluntarily by an unrestricted set of third-party contributors, it makes the following claim: *An OCSD process can itself be designed, with the principal objective of creating an environment for third-party innovation.* To support this claim the thesis outlines a conceptual framework to guide OCSD design. The framework includes a taxonomy of parameters and constraints relevant to opening design, a list of options within each taxonomic category, and three high level strategies found to recur as a function of sponsor goals and technological constraints. Finally, the thesis proposes a quantitative method, based on multidisciplinary modeling and pareto analysis, to design open standards within the context of one of the three strategies.

The research is carried out through a pragmatic blend of case studies and quantitative modeling. First, an in-depth, multi-discipline literature review synthesizes relevant taxonomic categories. Thirteen examples of OCSD spanning nine industries are then analyzed to define options within each taxonomic category. The case studies are also used to identify strategies for opening design based on correlations between OCSD options. The framework is validated and expanded through an in-depth case study of the opening of Very Large Scale Integration (VLSI) in the semi-conductor industry in the late 1970s. Finally, a quantitative method is developed to guide the design of open standards within one of the three strategies. These three contributions – the framework, correlated strategies, and quantitative method – are then applied to a particular biotechnology called microbial fuel cells.
Thesis Supervisor: Edward Crawley
Title: Ford Professor of Aeronautics and Astronautics and Engineering Systems

for Sarah

Acknowledgments

MIT is such a unique, fast-paced and vibrant place that a short time here surely lasts a lifetime. I have met and been helped by a tremendous number of amazing people during the last few years. In fact, I owe most of my education to meetings over coffee with friends, or discussions with professors who gave their time to help me along the way.

I would first like to thank my committee, without which the PhD would never have been possible. Ed Crawely, who despite having responsibility for thinking through the future of human spaceflight for the Obama Administration this summer, found time to keep my thinking clear and my writing tolerable. I want to thank Ken Oye for funding my work both through the Program on Emerging Technologies and the Synthetic Biology Engineering Research Center. Ken was particularly helpful in improving my methods and encouraging me to curb an instinct to continually broaden the research into multiple disciplines – perhaps because he understands the impulse himself. I want to thank Joel Moses, with whom I enjoyed many fascinating conversations. Joel engaged my ideas continually and helped me to better understand the implications of my findings. I know that with a few more months of research, he would have helped make this thesis better, and I look forward to future conversations about where it might go. I am very grateful to Tom Knight, who graciously lent his time and ideas about the broader implications of Synthetic Biology. I don't think I ever left Tom's office without being inspired about the field of Synthetic Biology or, for that matter, the world of ideas.

I owe a huge debt of gratitude to Oli de Weck, who supported me through both my masters degree, as a research scientist after the masters, and throughout the PhD. Professor de Weck has an amazing ability to combine theoretical insight with concrete suggestions for how to take ideas forward, and a genuine concern for his students lives and futures. I will always be grateful for his belief in my abilities, and his encouragement to pursue the things that inspired me, both in and out of MIT. Without this, I would never have been able to undertake the PhD, let alone complete it!

Many of the MIT staff were tremendous in helping me and my fellow students navigate The Institute. Kathi Brazil, Julie Finn, Beth Marois, Barbara DeLaBarre, Beth Milnes – you all made MIT manageable, and helped ease the stress of being a PhD student at MIT.

Along the way a number of MIT professors and other professionals generously provided their time and help. In the biological sciences, Drew Endy, Dianne Newman, Ron Weiss, Leonard Katz, all provided important insights into my thesis. More generally, Eric von Hippel, Chris Magee, Daniel Whitney, Frank Field, Merrit Roe Smith, Scott Mohr, Larry McCray, and a number of others were kind enough to sit down with me, often multiple times, to guide my thinking and provide advice. I thank you all very much for you encouragement and advice.

I am of course very thankful for the support of family, most of whom who may be as relieved as I am that finished. Sarah, my wife and the love of my life, was so supportive and understanding, I think she deserves the PhD as well! Mom, Gab, Amanda, Olivia, and Dad, listened to me, and helped me see the big picture as I moved towards the finish line. I also owe Nancy, Steve, and Jamie a huge debt of gratitude for your support and – of course - for letting me stay and write in the cottage in June of 2009. What an amazing experience to write a doctorate in the Madawaska Highlands.

Last, and of course not least, I have to thank all of my friends for being there for me and giving advice. You all made it fun. There are too many to list here, but you know who you are. At MIT, Travis, Chintan, Sid, Kathrine, Wilfried, Gergana, Ben Koo, Michel-Alexandre, Yves, David Broniatowski, Erica Gralla, Lara, Tristan, Daniel, Allesandro, Richard, Anthony, Soren, Kate Steel, Fabian Roth, Blandine, Gautum, Neely, and so many others (please forgive me for those accidentally left out!); outside of Boston, the Collegiate and Williams crew, Nate, LL, Bubb, Slaky, Golden, Peter, Christine Pace, Ted Fjallman, Pere "hanging 10" Roca Cusachs – and the rest of our fellow travelers. The list goes on but I'll leave it at that. Time to move onto the next chapter!

Table of Contents

List of Figures

List of Tables

Nomenclature

I = Operating current density (mA/cm^2)

L = Distance between electrodes (cm)

i_{ext} = exchange current density (mA/cm^2)

i_{lim} = Limiting current density (mA/cm^2)

R = gas constant (8.314 J/mol K)

T = cell operating temperature (K) – Room temp is 298K

F = Faraday's constant (9.64853e4 coulombs / mole of electrons)

n = Reaction Charge Constant (Mole e^- / Mole Fuel) (8 for Acetate)

α = Anodic Reaction Transfer Coefficient

V = Operating voltage (V)

E_{BOD} = Estimated Energy in One Gram of BOD = 4.0705 WH/Gram

V_n = Actual Open Circuit Voltage

V_T= Theoretical Open Circuit Voltage, OCV (Volts)

V_I = OCV Voltage loss due to current cross-over

V_B = OCV Voltage loss due to bacterial potential

C = Volumetric Cost ($/m^3$)

P_{cv} = Volumetric Power Density (W/m^3)

BOD_{cv}= Volumetric BOD Removal Rate (KG/m^3Day)

C_{Anc} = Ancillary Fixed Cost of Creating the Reactor

a = Cost of one square meter of membrane

W = Cost parameter dictated by membrane manufacturing process and materials

p_e = Resistivity of the electrolyte/water in (Ohm-CM)

R_{ext} = External resistance (Ohm)

A_A = Anode surface area (m^3)

u = Specific growth rate in unit cell mass per day

U_{max}= Maximum growth rate in unit cell mass per day day-1

Conc = Substrate Concentration in mg/L

K_c = Half Saturation constant in mg/L

Yield = Yield of bacterial mass per unit substrate consumed

X = Specific bacterial mass per electrode area (mg/cm2)

T_b = Biofilm Thickness (cm)

B_{ce} = Columbic Efficiency of the Bacteria (%)

$E_{current}$ = Number of moles of electrons converted to current per day

1 Introduction

1.1 Motivation

1.1.1 A new way of designing

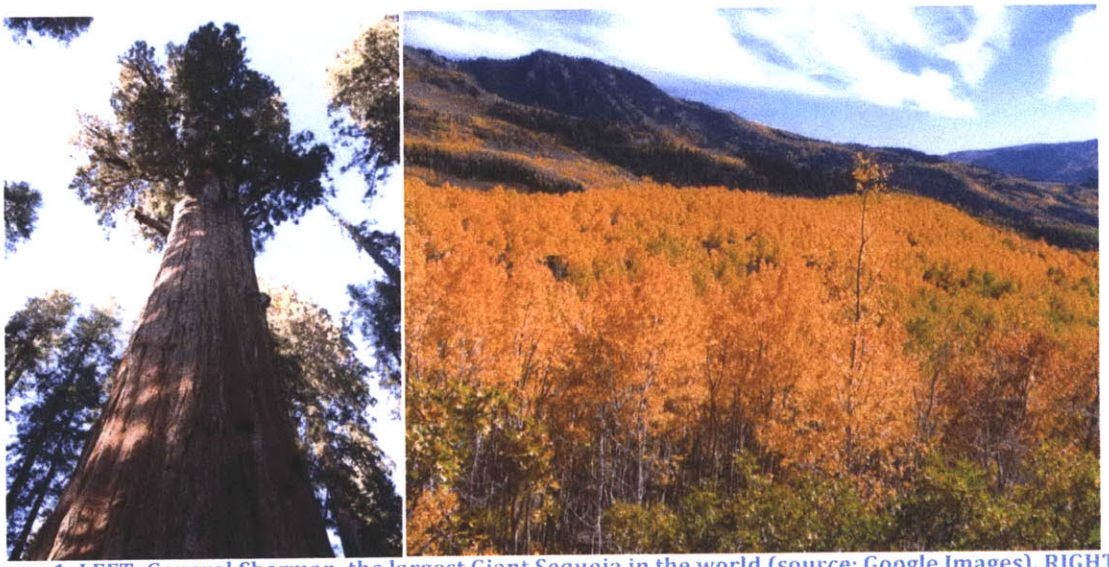


Figure 1: LEFT: General Sherman, the largest Giant Sequoia in the world (source: Google Images). RIGHT: The largest Aspen Grove in the world (source: Wikipedia).

An email circulating the Internet this year made a surprising point that the largest tree in the world is, not as many may assume, a Giant Sequoia like General Sherman (left). Instead, the largest single tree is an Aspen grove located in Fishlake National Forest, Utah. Why? Aspen groves are in fact single organisms sharing a common root structure. New trees crop up from the root wherever conditions are favorable. Size benefits both species, yet it is arrived at through fundamentally different strategies. The former makes one big bet, the latter thousands of small bets using shared nutrients.

The dichotomy frames well the broader motivation for this thesis. Across technology industries, and particularly at the cutting edge of bioengineering, a debate has been brewing about the proper balance between open and closed – sharing and hiding design information. As the Internet and related tools continue to drop the cost of collaboration through the "coasian floor," previous limits to the number of potential collaborators on a given project have been stripped away. The result is a vast new space of possibilities for organizing information-based design and production, with potentially profound implications for both our institutions and the goods we produce.

The broader public first began to be affected by mass collaboration through the emergence of usable open-source software tools in the late 1990s, such as the Firefox web browser and the GNU/Linux operating system. Yet, as a method of production, it is far from having run its course. There are now thousands of open source projects listed on www.sourceforge.net and the basic methods associated with very-large-scale collaboration are being explored throughout the media and even in traditionally heavy industries such as mining and aerospace. We still do not understand where this will likely go. As Clay Shirky has noted, "The increase in the power of both individuals and groups, outside traditional organizational structures, is unprecedented. Many institutions we rely on today will not survive this change without significant alteration, and the more an institution or industry relies on information as its core product, the greater and more complete that change will be." (Shirky 2008)

Despite the rise in its practical implementation, many practitioners in product design circles remain skeptical. Open strategies run contrary to established and well-supported economic and technical assumptions. Some note that opening the design of a complex system is often done as a last resort, only if a company is trailing a competitor or is otherwise forced to disclose proprietary information. At the most basic level, many question the economic sustainability of an innovation system based on giving away proprietary knowledge. How, they ask, could it ever be

profitable and sustainable to share something that likely reduces one's competitive advantage? What are the pathways to sustainable design and development in such a regime? It is not clear, in short, if the new emphasis on openness is a fad or enduring, whether it is software-specific or broader.

This thesis engages these questions within the context of recent attempts to open-source the design of biological systems in the field of synthetic biology. As a motivating consideration it is worth noting that, like nature in the example above, an economic system is rarely absolutist. Different environments call for different survival strategies. In Northern California large Sequoias (call them GE's or IBM's) will dominate. In other regions the more nimble Aspens that share critical nutrients and infrastructure will thrive. From this perspective, important questions emerge: in which economic climate and with which strategies will open design and production strategies take root and thrive? What are the limits to its application? How best might it be applied outside of software?

1.1.2 The Debate about Open Source in Synthetic Biology

The debate about opening design takes on special urgency in the field of bioengineering. Still in a period of rapid ferment and innovation, as of 2007 the biological industries produced 1% of GDP, growing at 20% annually (Carlson 2007). Rapid innovation in both enabling infrastructure and core scientific theory, together with the diversity of applications suggests it may become one of the core enabling technologies of the 21st century (Newcomb 2007). Subfields such as synthetic biology and Metabolic Engineering are currently striving to simplify the engineering processes in conscious emulation of computer engineering in the 1970s and 1980s (Endy 2005). This includes standardizing "components" (whether genetic or otherwise), reducing uncertainty associated with composition, increasing the use of simulation, and fostering the development of certain social and legal norms. If successful, these developments should create a paradigm shift – in the Kuhnian sense (Kuhn 1970) – in the design of complex biological systems.

Though the process of applying modern engineering practice in biotechnology is still nascent, important questions have already emerged with respect to opening innovation in the field. Lamenting the highly fragmented ownership structure of biological knowledge – both with respect to patent-thickets and industrial trade secrets – many in academia and some in industry have embraced the concept of "open source" design. Proponents of such an approach insist that if the foundational technologies associated with biological engineering are shared, innovation will increase dramatically to the benefit of private firms and society.

Opponents and skeptics of an open approach can be roughly divided into those who question the basic technological and economic feasibility, and those that are concerned about broader security and safety implications of distributing genetic engineering knowledge. For example, many claim that for the foreseeable future bioengineering remains more craft than scientific engineering (Hope 2004). They question whether "standards" make any sense with systems as exceedingly complex as biology. Of course, as a research program, synthetic biology is striving to address these very questions. As one researcher recently wrote: "The ability to quickly and reliably engineer many-component systems from libraries of standard interchangeable parts is one hallmark of modern technologies. Whether the apparent complexity of living systems will permit biological engineers to develop similar capabilities is a pressing research question." (Canton, Labno et al. 2008)

The high levels of technological uncertainty and capital expense lead many to insist that open design is not economically feasible in biotechnology. Josh Lerner and Jean Tirole articulate this concern well:

"Although some aspects of open source software collaboration (such as electronic information exchange across the world) could easily be duplicated, other aspects would be harder to emulate. Consider, for example, the case of biotechnology. It may be impossible to break up large projects into small manageable and independent modules and there may not be sufficiently sophisticated users who can customize the molecules to their own needs. The tasks that are involved in making the product available to the end user involve larger expenditures than simply providing consumer support and friendlier

user interfaces as in software. The costs of designing, testing, and seeking regulatory approval for a new drug are enormous…. the open source model may not easily be transposed to other industries, but further investigation is warranted." (Lerner and Tirole 2005)

Biotechnology may still be too immature or too costly as a field to apply open-source design principals. However, as the last line in their statement clarifies, this is a hypothesis warranting further research.

Economic and technical viability notwithstanding, many see other dangers arising from freely distributing bioengineering knowledge. Some in government fear that sharing a powerful enabling technology is a security threat, since the knowledge could be used to create weapons. Non-governmental organizations such as the "Erosion Technology and Concentration Group," or ETC, are very concerned about the ethical and safety implications of accelerating the creation of synthetic life forms.

Addressing these kinds of questions is important. Bioengineering is likely to have a tremendous impact on our society, due both to the range of industries it will affect, and the way in which it is likely to impact production itself. As of today, the industry produces therapeutics, tissues, medical devices, crops, fuels, specialty chemicals, even plastics. Trends in these end-product categories are potentially explosive. For example, the USDA projects that bio-based products will grow from 2% to fully 22% of the $1.3 Trillion global chemicals market by 2025 (USDA 2008). This is due in part to advances in bioengineering and to the comparatively benign manufacturing and energy requirements associated with the fermentation of complex products. New product-classes will emerge as the technology progresses. Personalized medicine, biosensors, industrial chemicals, microbial fuel cells, cellular computers – the range is extraordinarily broad.

Beyond broadening *what* we produce, biotechnology has the potential to change the means of production itself. Engineers can increasingly sequence (read) DNA into a

computer and synthesize (write) it reliably and cheaply. This makes DNA fungible with information and turns biological design into information-based production—exactly the domain in which the battles of open versus closed rage the strongest. Information-based production is prone to network affects, winner-take all dynamics, and a number of other economic particularities caused by the fact that it is expensive to develop but cheap to replicate (Shapiro and Varian 1999). As bioengineering becomes information-based production there may be severe changes to the economics of physical production. If open source methods can be best applied within information intensive industries, they may well have a particular impact in the domain of biotechnology.

1.2 General Objectives

This thesis has the general objective of exploring the potential for open design and innovation within the context of synthetic biology. The general objective faces two major classes of questions. First, how is the engineering of biotechnology likely to develop in the coming years – will standards-oriented, library-based design be successful? Or will more integrated, capital-intensive, industrial processes continue to predominate? Second, how can open innovation be employed as a function of this end-state?

Given the great amount of uncertainty surrounding both sets of questions, it is currently difficult to fully address both without resorting to speculation. Therefore, this thesis focuses largely on questions of opening innovation, using biotechnology as one particular potential domain of application. The goal is a synthetic, cross industry framework, to clarify options and guide strategy for opening design. This framework is then applied and evaluated within the context of a particular biotechnology called microbial fuel cells.

Two basic factors motivated the choice of this refined objective. First, questions regarding the effectiveness of modern engineering concepts in bioengineering can

be solved almost solely through practical experimentation. Research efforts in labs around the world, together with the help of organizations such as the synthetic biology Research Consortium, will ultimately define the field. Second, as a doctoral thesis in the emerging field of Engineering Systems, this dissertation has prescriptive as well as descriptive goals. More than simply describing what form an open innovation regime might take within the biotechnology domain, the goal was to develop tools, analyses, and ways of thinking that can guide the design of strategies for opening design.

The goal of developing an actionable framework for opening design and exploring its application to biotechnology is still, of course, very broad. Before articulating the specific objectives and approach, it will be helpful to provide more background on the subject, and clarify definitions.

1.3 Defining Open Collaborative Systems Development

What exactly is meant by the phrases "open innovation" or "open design?" There is much confusion in the literature on this point. The concept of "opening" design or innovation generally refers to sharing what might otherwise be proprietary design information in an effort to co-develop products, systems or services together with third parties. However, the term must be defined more precisely for research purposes.

Three of the most important literatures in this area include the notion of "user innovation" (Von Hippel 2005), "open innovation" (Chesbrough, Vanhaverbeke et al. 2006), and "collaborative systems" (Maier and Rechtin 2002). *User innovation* generally refers to the free exchange of design information by users of a technology; *open innovation* has been used to refer to IP-intensive business models involving spin-offs or spin-ins to large corporations; *collaborative systems development* derives from the systems engineering literature, and involves opening interface

standards for co-development of complex products.[1] These three perspectives share some important attributes. Most importantly, all three make clear that open design is a *process for designing and developing* systems by sharing certain kinds of information with third party collaborators. However, the literatures differ on how this information is shared and who shares it. The three literatures have, in short, differing conceptions of the word "open."

This thesis examines the problem from the perspective of a systems architect developing an open, collaborative development process. It therefore builds upon a definition of open system development articulated by Maier and Rechtin in their book The Art of Systems Architecture (Maier and Rechtin 2002).[2] As they write:

"Many systems are not under central control, either in their conception, their development, or in their operation...these systems are collaborative in the sense that they are assembled and operate through the voluntary choices of the participants, not through the dictates of an individual client. These systems are built and operated only through a collaborative process." (Maier and Rechtin 2002)

For Maier and Rechtin, then, there is a class of systems of such complexity that they can only be effectively developed through bottom-up, collaborative processes. Examples of such systems include The Internet, intelligent transportation systems, and joint air defense systems (Figure 2). According to Maier and Rechtin, the goal for designing such systems can be construed as "maximizing investment opportunities" for third parties, rather than maximizing benefit/cost (Maier and Rechtin 2002).

---

[1] "Open source software" can be considered a specific case of one or more of these general definitions, as described in more depth in the literature review.

[2] Due to the flurry of research on this topic, important insights based on differing perspectives and definitions have been developed. The literature review in Chapter 2 therfore grounds the discussion by describing the basic process of open source software design, and also reviews the differing schools of research on open innovation.

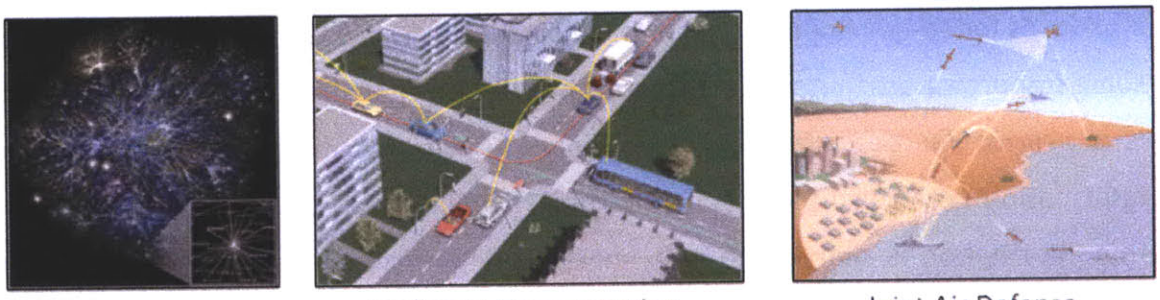


Figure 2: Examples of collaborative systems.

Maier and Rechtin classify systems as collaborative based on the way in which *components* are designed and produced – whether they are created voluntarily, by different entities, and whether they are managed for their own purposes (Maier and Rechtin 2002). However, a collaborative development process may or may not be *open*. This point was clarified in a recent survey article by Gary Pisano and Roberto Verganti (Pisano and Verganti 2008). Pisano and Verganti distinguish open collaboration from closed collaboration based on *restrictions to the set of potential contributors*. An open and collaboration regime is one in which any entity could theoretically contribute to the design. A closed collaboration regime is one in which a select few can contribute. Figure 3 provides examples of each kind of process.

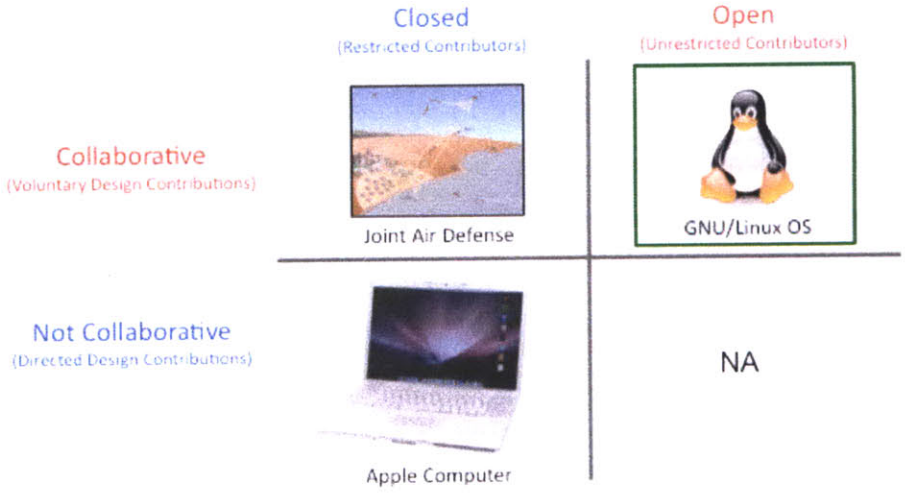


Figure 3: Two dimensions to collaborative production. Collaboration refers to whether or not the design work is directed from the top, or voluntarily contributed from below. Opennes refers to whether or not any developer can contribute to the design process.

To be clear, "open" in Figure 3 is used in a different way than in the literatures deriving from Chesbrough and Von Hippel. For Chesbrough, "open innovation"

implies the use of knowledge across firm boundaries. While for Von Hippel "user innovation" and "collective design" are processes in which designs are partly or fully free from intellectual property restrictions. In the figure above, open implies no restrictions on the set of potential contributors.

This thesis uses Pisano's definition of openness. The choice was made for a number of reasons. First, it is unambiguous. Second, it corresponds more directly to the constraints on the kinds of systems categorized by Maier and Rechtin as "collaborative." The Internet, for example, is a network defined in part by the fact that any computer can connect and contribute content. Third, the definition encompasses the current vision for open design in synthetic biology.

Combining and modifying the definitions by Maier, Rechtin, and Pisano, this thesis defines "open collaborative system development" (OCSD) as the creation of a *complex system or database in which some or all of the component designs:*

1. Are created by an unrestricted set of potential developers
2. Based on the developers' own initiative (e.g. voluntarily)

This definition creates a three-part test concerning whether or not a development process occurs through OCSD. First, do the designs contribute to a more complex system or database? That is, do third-parties design *the system components*? This distinction is made because the cases in which third-parties voluntarily create the entire system are actually very similar to the open market. The concern, in this thesis, is with the creation of complex systems via OCSD.

The second test, based on the definition above, involves whether components designed based on the developers' own initiative? This corresponds to the Y-axis in Figure 3 above. The criteria is meant to distinguish between processes in which design work is clearly directed from a centralized source, and design work which emerges un-directed from a pool of collaborators. This distinction is made because,

as emphasized below, an important benefit of OCSD processes involves the removal of overhead needed to coordinate the logistics of design work.[3]

The final test to determine whether a system is designed through an OCSD process involves whether the developer pool is restricted or not. The meaning of "restricted" was clarified in the paragraphs above.

As Figure 3 above makes clear, the three-part definition clearly delineates OCSD process from non-OCSD process. For example, GNU/Linux was developed through an OCSD process. The individual modules of GNU, including the Linux kernel, were designed and managed voluntarily by an unrestricted set of individuals. Similarly, the registry of standardized biological parts fits this definition. It is a storehouse of DNA parts, devices and systems (designs) that have been created voluntarily by an unrestricted set of diverse teams, based largely on an a decision to participate in the iGEM competition.[4]

The definition also excludes certain development processes. The Joint Air Defense system, for example, may have been developed through a collaborative process, but it was not developed through an *open*, collaborative process. Most firms and individuals are restricted from contributing component designs to the Joint Air Defense System. Similarly, Apple computer developed the macbook pro through a process that is both closed and non-collaborative. Design work, in this case, was carried out by Apple. Some component designs were carried out by sub-contracted firms, but only in restricted arrangements and through Apple's explicit design direction. Table 1 below tests additional development and acquisition processes against the three-part definition.

Table 1: The three-part test for OCSD administered to a range of development and system aquisition processes. A check indicates the process passes the test; a red cross indicates test failure. In this table, only GNU/Linux fits the definition. The Boeing 787 includes both a check and a cross regarding the direction of design works. This is because, while the 787 involved unprecendented design freedom to its

[3] As discussed below, this benefit is accompanied by other potential problemsor costs thath must be addressed.

[4] Of course anyone, besides iGEM teams, can also contribute to the registry.

sub-contractors, specific contracters were stil directed to perform specific design tasks.

	Space Shuttle Subcontractors	SBIR Small Business Innovation Projects	Being 787 Component Designs	GNU/Linux Operating System
Components of a larger system?	✓	✗	✓	✓
Undirected design work?	✗	✓	✗*	✓
Unrestricted set of developers?	✗	✓	✗	✓

Additional attributes of the definition are worth emphasizing. First, an OCSD is not by definition free of intellectual property restrictions. Second, an OCSD may or may not have a precise end-goal. Third, the definition above clarifies that OCSD is a process for producing designs. Like, the "waterfall method" in software or the "V" method in traditional systems engineering, an OCSD process should be considered a way of bringing about new systems. As such, the process can be consciously designed by a system architect, depending on the goals for the system and constraints associated with the technology.

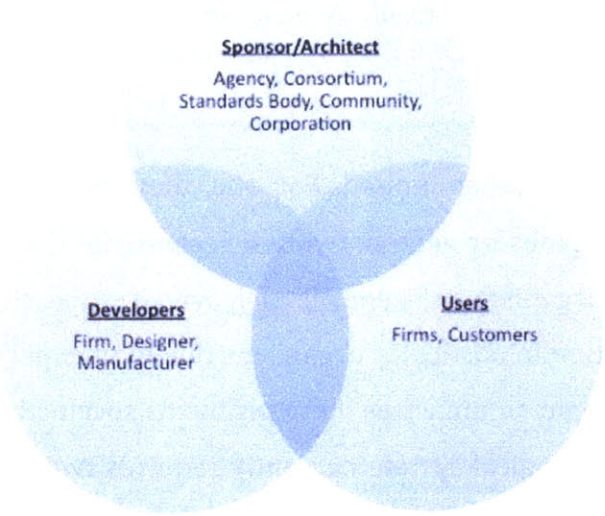


Figure 4: Stakeholders in collaborative system development. This diagram represents the overlap of sets of stakeholders, not necessarily their structure.

Finally, as a development process, OCSD will have distinct stakeholders (Figure 4). These include sponsors or architects, users of the system, and developers of the sub-systems (Maier and Rechtin 2002). It is important to emphasize that these classes of

stakeholders are not mutually exclusive. For example, the sponsors of the regime might also be users or developers.

1.4 Synthetic Biology

Why would an OCSD process be useful for designing biological systems? An answer to this question requires some background on the discipline of synthetic biology. As a field, synthetic biology might be labeled Bioengineering 2.0. It is a conscious attempt to overcome the limitations of existing bioengineering practice through new tools, techniques, and a conceptual framework derived from modern scientific engineering.

Until recently, the term biological engineering was most closely associated with recombinant DNA techniques that expressed single genes in host bacteria. This was accomplished using a variety of methods to isolate, copy, and insert genes of interest into host bacteria for expression. The techniques improved continuously over the past decades, and have been used to develop several important chemicals and products.

Yet these methods are rather limited. For one, they do not contemplate creating systems of multiple genes or gene networks. Second, the details of each technique are specific to each organism and gene. Finally, even if successfully expressed, target genes may not function in particular organisms due to unexpected interactions with the host. Dr. Tom Knight summarizes these problems succinctly by noting that every design change in a biological system currently requires two experiments – the first establishes whether the change can be made in the first place, the second determines whether the design change works.

In their quest for control and reproducibility in the biological context, synthetic biologists have imported a number of concepts developed in established engineering fields. They envision a day when, as in the electronics industry, a

number of "standard" parts can be assembled by practitioners to build devices that can, in turn, be combined to form complex systems. This requires formalization of three basic systems engineering concepts in the context of biology: standardization, modularity, and abstraction of modular elements within a hierarchy (Endy 2005). Figure 5 compares the old paradigm to synthetic biology.

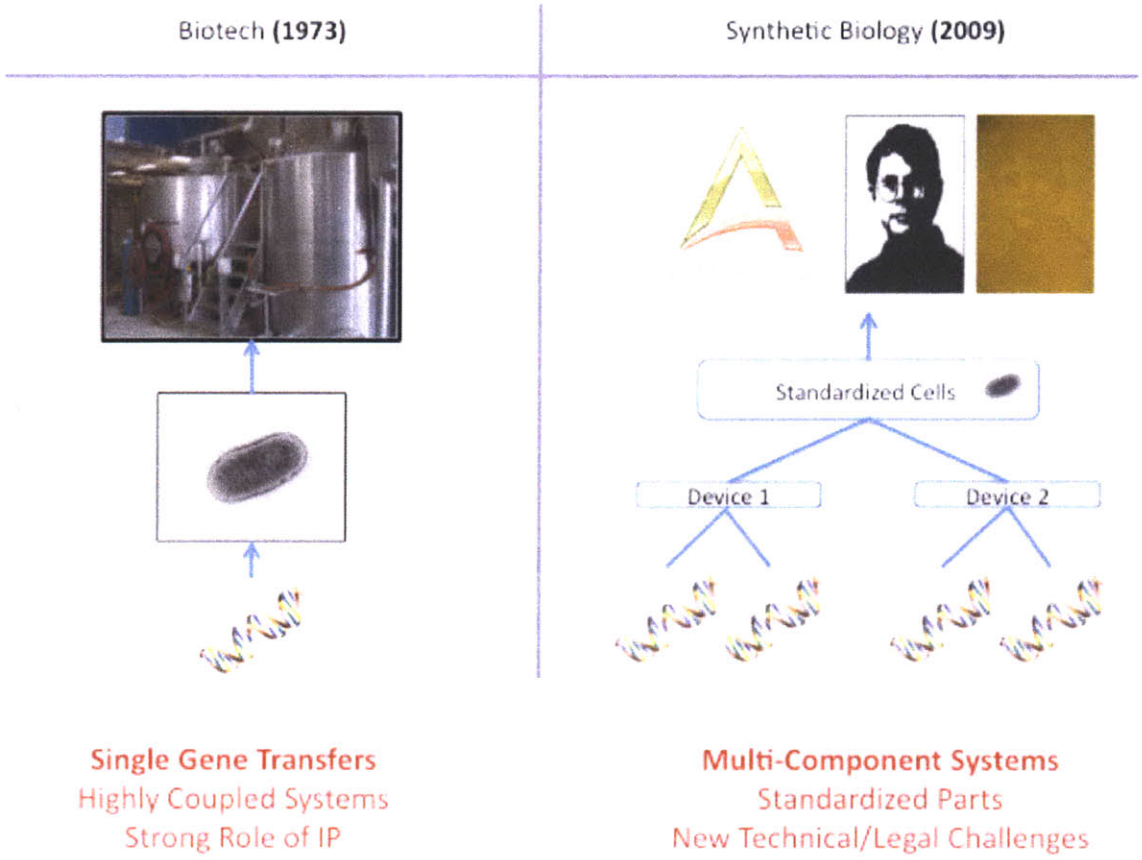


Figure 5: Traditional bioengineering versus synthetic biology. Synthetic biology results in a wide range of end applications including bacterial photographs, multi-component bio-fuel platforms, and others.

As the basic information-carrying unit and interface between high-level design and fabrication, DNA within this paradigm is equivalent to source-code in software. Therefore, from a technical perspective, the rapid prototyping of new biological designs requires three capabilities: 1. Reading DNA (called "sequencing") 2. Writing DNA (called "synthesis") 3. Inserting DNA into cells such that it can be translated and used (called "expression"). Very rapid advances are being made in all three domains. For example, Figure 6 below illustrates that the cost of DNA sequencing and synthesis is falling exponentially.

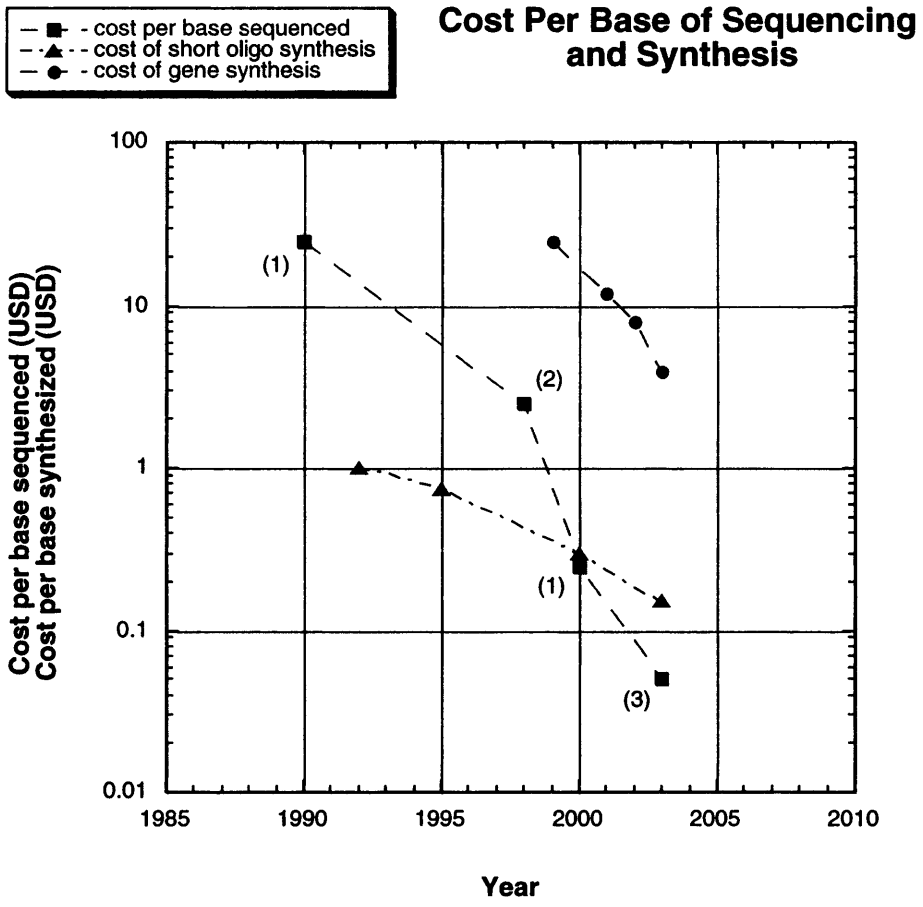


Figure 6: The falling cost of DNA Synthesis and Sequencing (courtesy: (Carlson 2003)).

As the cost of sequencing and synthesizing DNA drops, the limiting factors in synthetic biology's vision therefore becomes the simplicity and reliability of expression (inserting new gene-networks into host cells) and, importantly, the transaction costs associated with using DNA. Reliable expression is a tremendous challenge. Multiple efforts are underway to solve it by developing standard assembly techniques at various levels of abstraction. These include open standards for physical assembly, standards for functional assembly, and standard host-cells called "chassis." To date, however, unpredictable interactions between foreign DNA and host organisms have thwarted reliable expression.

The second limiting factor in synthetic biology's vision – high transaction costs associated with using DNA and other biological elements – touches directly on the issue of openness because high rates of patenting restrict developers' freedom to

design certain systems. It is well known that the number of patents within biotechnology has greatly increased in recent years. Figure 7 illustrates this point by comparing new patent filings between 1990 and 2008 in two biotech classes and two non-biotech classes.

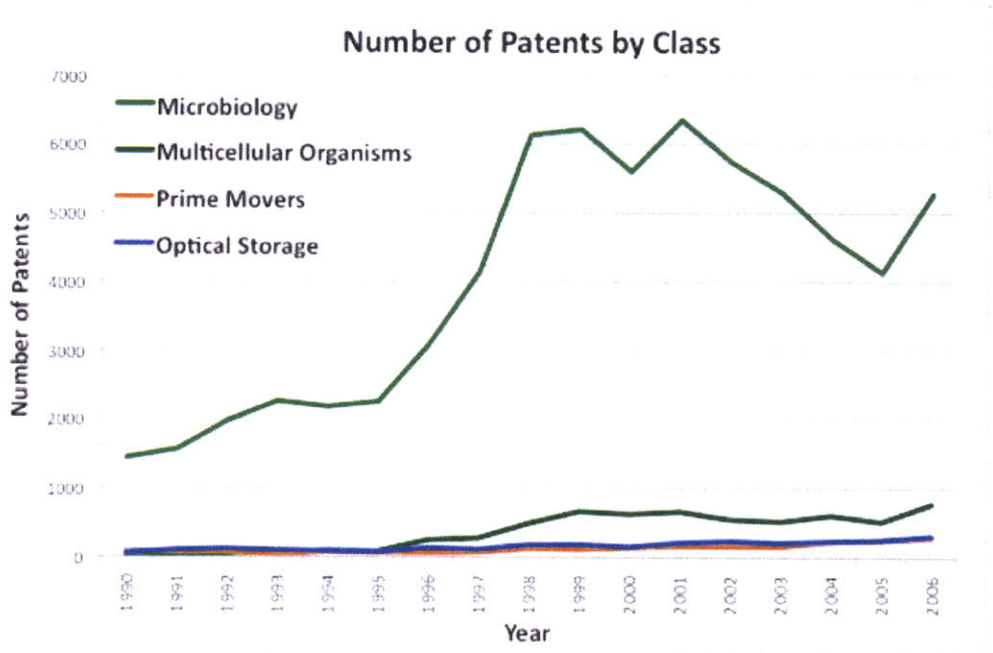


Figure 7: Comparing new patents by class. (source: USPTO)

The figure shows a very large disparity between the number of patents filed annually within and without biotech. In 2002, for example, there were 6000 patents filed under the class "Microbes and Microbiology" but only 300 filed under "Optical Storage." The disparity results from more than increased innovation rates. From a legal perspective, DNA sequences are considered patentable for particular uses, even if they have not been invented. This leads to a tremendous amount of "strategic patenting" which can create "patent thickets" that greatly increase the transaction costs associated with innovation (Oye and Wellhausen 2009). Even in the academic context, transaction costs associated with investigating new genes are increased by cumbersome materials transfer agreements (MTAs).

These issues thus formalize how an open innovation paradigm may benefit synthetic biology and metabolic engineering as fields. The cost of sequencing and synthesizing DNA is plummeting. Despite the exceeding complexity of biology, the

use of open assembly standards and modular design principals are moving bioengineering from a craft-like activity towards information-based production. Yet, strategic patenting of DNA and cumbersome MTAs slow innovation by raising transaction costs. Given that there are a limited number of uses for particular genes, the problem is thus probably more acute than copyright and ownership of source code in software. The question remains whether these problems can and should be resolved by employing open, collaborative innovation processes? If so, how best should an open platform for biological engineering be created?

Some of the questions about open-sourcing biotechnology are being addressed practically through new institutional arrangements. *The Registry of Standardized Biological Parts* (http://partsregistry.org) was developed at MIT to catalogue and house freely available DNA parts, systems, devices and chassis. *Synberc* (http://www.synberc.org) is an NSF-funded organization with the goal of developing infrastructure for synthetic biology. It is developing a number of test-beds designed to demonstrate the effectiveness of standard biological parts. *Open Wetware* (http://openwetware.org) is an online project designed to help spread information concerning synthetic biology practices and protocols. The *Biobricks Foundation* (http://bbf.openwetware.org) has been established recently to develop standards for creating and sharing DNA parts. The *International Genetically Engineered Machines Competition* (http://2009.igem.org) encourages student teams from around the world to use this registry to create and share novel genetic designs. While these efforts are positive, they do not conform to a broader strategy or framework for developing open, collaborative technology platforms for a simple reason: no framework or broader set of strategies yet exist.

1.5 Specific Questions and Thesis Statement

The goal of this thesis is to develop a broad framework for designing an open collaborative systems development process, and to validate its application within the domain of biotechnology. The definitions and background provided in the

previous sections enable a more precise framing of the thesis questions and objectives. As Maier and Rechtin emphasize, our current ability to design collaborative systems is very limited. A simple illustration of this problem is that no clear lexicon exists to describe and design such systems (Maier and Rechtin 2002). Further, there is no framework to guide such an approach. The specific challenge can therefore be articulated as follows:

1. The registry of standardized biological parts is an attempt to create an open, collaborative, system development process (OCSD).

2. There are a number of challenges associated with creating OCSDs:
    a. There is no clear lexicon to discuss OCSD
    b. There is no integrated framework to design OCSD
    c. There has been little analysis of how technology type and system architecture relate to OCSD.

The thesis makes the following claim, to be validated through research:

1. An OCSD process can itself be designed.

2. Specifically, OCSD can be treated as a multi-objective design problem where the principal goal is ***to design an environment for third-party innovation.***

3. One can develop an integrated conceptual framework with which to craft strategy and evaluate outcomes for OCSD which includes:
    a. A taxonomy of conceptual building blocks
    b. Options within each taxonomic category
    c. Integrated strategic options

4. The framework can clarify constraints and opportunities in the development of open platforms in synthetic biology.

1.6 Contributions and Argument

This thesis argues that the process of developing a new system by OCSD can itself be designed. It supports this claim by making contributions at both strategic and technical levels. At the strategic level, a framework is created which includes a taxonomy of "inputs," "design variables," and "results" associated with creating an OCSD. Further, three distinct strategies – called Incremental Development,

Architectural Development, and Database Development - are found to recur within this framework. Finally, at a technical level, a quantitative method is developed to guide the design of open standards within the context the Incremental Development strategy. The quantitative method, based on multidisciplinary design analysis, operationalizes the design of open standards through mathematical modeling. It is applied to a biotechnology called Microbial Fuel Cells (MFCs).

1.6.1 Framework: Taxonomy and Options

The proposed framework consists of eight taxonomic categories that clarify what kinds of options are most important if one wishes to design an OCSD (Figure 8). Inputs (red) include the goals of the sponsor and constraints associated with the technology whose open development is being planned. Design options (green) include legal arrangements governing intellectual property, the kind of standards used, and the nature of what is shared between co-developers. Results or outputs (blue) include the kind of innovation that sponsors seek, the nature of incentives that will predominate in the process, and the economic strategies employed by system designers and developers. The thesis identifies a list of options for each category.

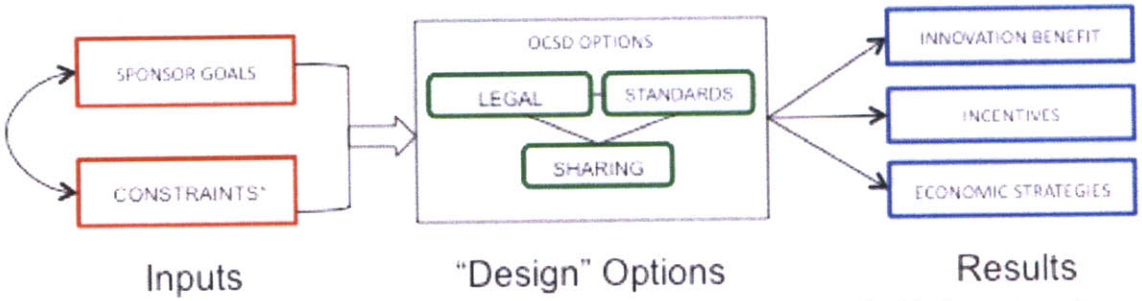


Figure 8: Taxonomic categories in the framework. Constraints are associated with the system whose development is being planned.

1.6.2 Innovation for Performance versus Functionality in Open Regimes

Using case studies of fourteen regimes that fit the OCSD definition provided above (Table 2 below), this thesis argues that several important distinctions about OCSD processes together define three distinct strategies. From a systems engineering perspective, one of the most important distinctions involves whether sponsors seek

to improve system performance and reduce technical uncertainty, or whether they seek to identify new functionality.

Table 2: Thirteen cases analyzed in Chapter 3, as well as the detailed VLSI case from Chapter 4. Orange cases share performance-constrained designs. Green share databases. Blue share standards and some design patterns in order to find new end-uses.

	Case Name	Sponsor Type	System Knowledge	Market Knowledge	Primary Development Goal	What does the System Transform?	What is Open?
1	Cleveland	User	Low	High	Better Performance	Energy	Full Design
2	Cornwall	User	Low	High	Better Performance	Energy	Full Design
3	Bessemer	User	Low	High	Better Performance	Energy	Full Design
4	NASA Clickworkers	User	Low	High	Better Performance	NA (Database)	Data
5	SETI Database	User	Low	High	Better Performance	NA (Database)	Data
6	Goldcorp Mining Co	Developer	Low	High	Better Performance	NA (Database)	Data
7	Alexa Search	Developer	Low	High	Better Performance	NA (Database)	Data
8	SNP	Developer	Low	High	Better Performance	NA (Database)	Data
9	Chongqing	Developer	High	Low	**New End-Uses**	**Energy**	Interfaces & Modules
10	Red Hat	Developer	High	Low	New End-Uses	Information	**Full Design**
11	Google Maps	Developer	High	Low	New End-Uses	Information	Interfaces & Modules
12	Ebay Developers	Developer	High	Low	New End-Uses	Information	Interfaces & Modules
13	Amazon Developers	Developer	High	Low	New End-Uses	Information	Interfaces & Modules
14	VLSI	Developer	High	Low	New End-Uses	Information (Hardware)	Interfaces & Modules

While system designers often seek a mixture of improved performance and/or new functionality when designing new products, it was found that successful OCSD Sponsors have the goal of either new functionality or improved performance, but rarely both. Further, the basic dichotomy between performance and function correlates to other aspects of OCSDs (see Table 2). For example, in cases where systems are performance-constrained, opening tends to be used by sponsors to *improve their knowledge of a system* by sharing *complete system designs.* These cases are highlighted in orange and green in the table above. In cases where performance is less of a concern, sponsors tend to use OCSD processes to enlist third party developers to help *diversify end-applications.* These two patterns are highlighted in blue and orange, respectively.

The two goals are shaped by and shape other factors in sustainable open development processes. From a technological perspective, they correlate largely, though not completely, to whether the system being designed *transforms energy or information*. For example, case 1 in the table above involves open collaborative development of blast furnaces. Blast furnaces are an energetic technology because they convert coal into heat, which is used to change iron ore into pig iron. The OCSD in this case was sponsored by a number of mine owners who had the objective of improving furnace performance. This was therefore a case in which OCSD Sponsor sought to improve performance of an energetic technology by sharing complete designs.

Conversly, new functionality was almost only sought by OCSD Sponsors when the system being designed transformed information. For example, the publication of the Google Maps API was a conscious effort to encourage web-service developers to find new end-uses for Google's mapping technology.

However, as articulated in more detail in chapter 3, there are important exceptions to this energy-information correlation. A number of sponsors in diverse industries shared databases, which were used to develop products that may or may not be information intensive. For example, a mining company called Goldcorp shared data associated with its mine. In these kinds of cases the shared element – data – is information. Yet many of the ancillary correlations described in Table 2 – including the level of system knowledge, level of end-use knowledge, and primary development goal – are similar to the energy-intensive cases highlighted in orange. Sharing databases is therefore defined as a distinct strategy in this thesis (green).

A second exception to the energy-information dichotomy is highlighted in the red boxes in the table above. One case (Chongqing) involves an energetic technology (motorcycles), yet the OCSD sponsors sought to diversifying end-uses. The exception suggests that the distinction between whether OCSD sponsors seek improved performance versus new functionality is more important than whether

the system process energy or information. That is, it is more likely that the energy-information correlation is a product of the fact that energetic systems tend to be performance constrained.

1.6.3 Three Strategies for Designing an OCSD Regime

These distinctions correlate to a range of non-technological attributes and thus frame the following three strategies for designing an OCSD: (1) Incremental Development (1.A) Database Development (2) Architectural Development (Table 3). Database development is labeled (1.A) because it is considered a subset of the Incremental Development Strategy.

Table 3: Three strategies for designing OCSD.

		Suggested Inputs			Suggested Decisions		Likely Result
	Name	**Typical Sponsors**	**System Knowledge**	**Market/Need Knowledge**	**Principal Elements Shared**	**Important Standards**	**Type of Innovation**
1	**Incremental Development**	Users	Low	High	Entire Designs	Measurement, Description	Incremental
1.A	**Database Development**	Users or Developers	Low	High	Data	Measurement, Data Formats	Incremental Pre-Competitive Research
2	**Architectural Development**	Developers	High	Low	Design Patterns, Interfaces	Vertical Standards, Data Formats	Architectural

These three strategies provide general constraints on OCSD design decisions. For example, *Incremental Development* is generally employed by *users* of a poorly understood or highly complex technology with very specific market objectives. These user-sponsors share entire designs using a standard description. The *Architectural Development* strategy is generally employed by for-profit developers who understand a technology very well but do not have firm knowledge of potential markets or needs. In these cases, part of the design (e.g. a design pattern) can be shared using interface standards. As described in more detail in this thesis, these strategies can thus be used by potential sponsors of an OCSD regime to better

understand how to craft the regime itself – for example, what to share based on their goals.

However, it is important to emphasize that these strategies do not fully encompass the range of options identified in the broader proposed framework. This is because some options are not necessarily constrainted by the choice of one or another strategy. For example, the broader framework makes a distinction whether OCSD sponsors share "horizontal interface standards" or "vertical interface standards." The former connect sub-systems *within* a given layer in the value change. The latter provide connections between layers of the value chain or layers of abstraction in a complex design. For example, connections between hardware subsystems in personal computers are horizontal. Application programming interfaces (APIs) between an operating system and end-applications are vertical.[5] This decision can be made within the Architectural Development strategy, and thus constitutes a set of options at a lower level of granularity than the strategies themselves.

This thesis argues that vertical interfaces are more important than horizontal interfaces, at least for the purpose of creating an OCSD. Horizontal standards divide design labor. Vertical standards enable *portability* of higher-level designs across multiple lower-level designs and thereby decouple the search for new end-applications from the design of underlying systems. An example of vertical decoupling in the biological industry would be the interface between chassis microbes and "bioreactors," or between DNA parts and chassis microbes (Figure 9). These distinctions are clarified as they apply to the biotechnology domain in Chapter 6.

[5] As discussed in more detail in Chapters 3,4 and 5, designers sometimes have discretion with respect to what constitutes a given layer in a value chain.

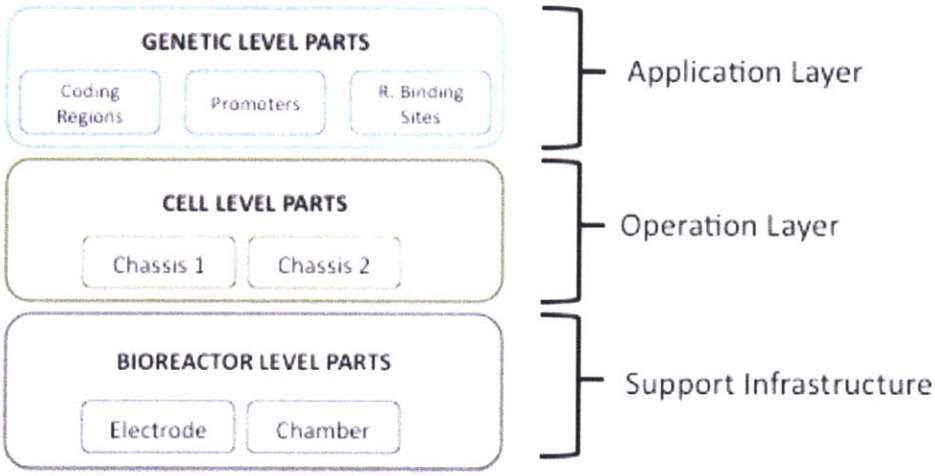


Figure 9: One conceptualization of the vertically decoupled biotechnology value chain. Coding regions, promoters, and R. binding sites are parts made out of DNA. Chassis are the synthetic biology term for standardized microbial cell line. Bioreactors provide the environment in which microbes can grow.

The three strategies presented in this thesis therefore synthesize potential sponsor objectives with architectural properties, kinds of sharing, and kinds of standards. The broader framework also proposes a set of economic strategies that developers are likely to pursue within an open regime, as well as a set of legal mechanisms that appear to predominate. However, these two taxonomic categories – legal mechanisms and developer strategies – are not found to correlate significantly with sponsor goals, and system constraints, at least for the cases analyzed.

1.6.4 A Quantitative Method for Designing Standards in Open Regimes

Beyond providing strategic guidance for OCSD sponsors, this thesis develops a quantitative method to design open standards within the context of the Incremental Development strategy. It then applies this method to the design of a biotechnology called microbial fuel cells. The method has four steps:

1. Create a multidisciplinary model of the technology
2. Identify feasible bounds on relevant parameters and create a pareto plot
3. Identify and visualize sub-sets within the feasible set of designs that correspond to standardized design variables
4. Calculate losses against objectives associated with constraining the set of designs to the standardized subset

Because design variables are bound using physical constraints – rather than knowledge of internal system function or internal constraints – the method can be

employed without significant knowledge of potential design solutions. It can therefore be employed within the Incremental Development strategy outlined in the broader strategic framework, in which a sponsor with low system knowledge seeks to develop a technology to address specific market needs. Results can guide standard setting.

1.7 Approach and Thesis Roadmap

Given the breadth of the topic, the proposed research uses a pragmatic, mixed-methods approach involving what Creswell has defined as "sequential mixed methods" (Creswell 2003).

Chapter 2 provides a wide ranging, multidisciplinary survey of the literature on open innovation, open source, and user innovation, in order to develop a preliminary conceptual framework and develops taxonomic categories for the design of an OCSD.

Chapter 3 utilizes a grounded theory to analyze fourteen previously identified examples of OCSD processes with the goal of further developing the proposed framework and identifying correlations among drivers of the OCSD. In particular, building on taxonomic gaps identified in Chapter 2, it identifies correlations between properties of the operating environment, technological constraints, sponsor goals, and the type of innovation that will likely take place within an OCSD. These correlations are used to define three distinct strategies for designing OCSD, as a function of sponsor goals and technological constraints.

Chapter 4 validates the framework through an in depth case study of the opening of Very Large Scale Integration (VLSI) in the semi-conductor industry in the late 1970s and early 1980s. The chapter validates correlations identified in Chapter 3 and further clarifies distinctions regarding technological constraints, standards, and

openness. Further, the chapter describes socio-cultural challenges associated with shifting from closed organizations to an OCSD.

Chapter 5 applies the proposed framework within the context of one particular biotechnology called Microbial Fuel Cells. This includes a description of the basic technology, and an articulation of how two of the three strategies within the framework might be applied within the field. Qualitative benefits and costs and bio-specific challenges for each strategy are identified.

Chapter 6 then builds on Chapter 5 to develop a simple, quantitative method to explore the potential performance losses associated with standardizing and opening aspects of Microbial Fuel Cell technology. The method is based on multidisciplinary modeling and pareto analysis. It enables both visualization of the impact of standardization in the metric space including calculation of potential performance losses of future systems designed with the standards and identification of optimal standards. Results from the method can be used to evaluate the potential implications of opening and standardized individual elements of the system. This method is applied to a multidisciplinary model of a microbial fuel cell. It utilizes one of the three proposed strategies within the broader OCSD.

Finally, Chapter 7 provides a conclusion and discussion of the thesis including a summary of the contributions, implications for synthetic biology, and potential directions for future research.

2 Literature Synthesis and Framework Initiation

"The developer who uses only his or her own brain in a closed project is going to fall behind the developer who knows how to create an open, evolutionary context in which feedback exploring the design space, code contributions, bug spotting, and other improvements come from hundreds (perhaps thousands) of people."

- *Eric Raymond, The Cathedral and the Bazaar*

2.1 Introduction

Open Collaborative System Development (OCSD) can be employed as a strategy for technological innovation. This thesis argues that the central problem associated with sponsoring an OCSD is the design of an environment in which third parties can innovate. The problem of designing such an environment can be formalized using the language of design optimization, with defined objective functions, constraints, design variables, and parameters. One of the key challenges with treating the creation of an OCSD is the lack of a clear lexicon associated with the process. More generally, currently there is no framework for organizing relevant design variables and parameters.

These high level gaps exist despite a substantial literature on open innovation and open source, and substantial practical work to open the design of complex biological systems. Many key concepts have been elucidated through empirical and theoretical work in a range of disciplines. A first step towards creating a coherent framework for OCSD design is therefore to assemble these diverse concepts into a taxonomy.

This chapter presents a wide ranging analysis of the literature on open innovation and open source development, with the goal of initiating a framework and specifying gaps within it. The review covers five essential areas: (1) Incentives (2)

Legal and Intellectual Property Options (3) The Organization of Design Activity (4) Systems Architecture and Modularity and (5) Developer Strategies. The purpose of this review is not to address every study in each domain, since that would number in the hundreds, if not thousands of papers. Rather, the review examines the most important works in each domain in order to extract relevant taxonomic categories. Before delving into these strategies, a basic overview of the work and questions on open innovation across technological domains is presented.

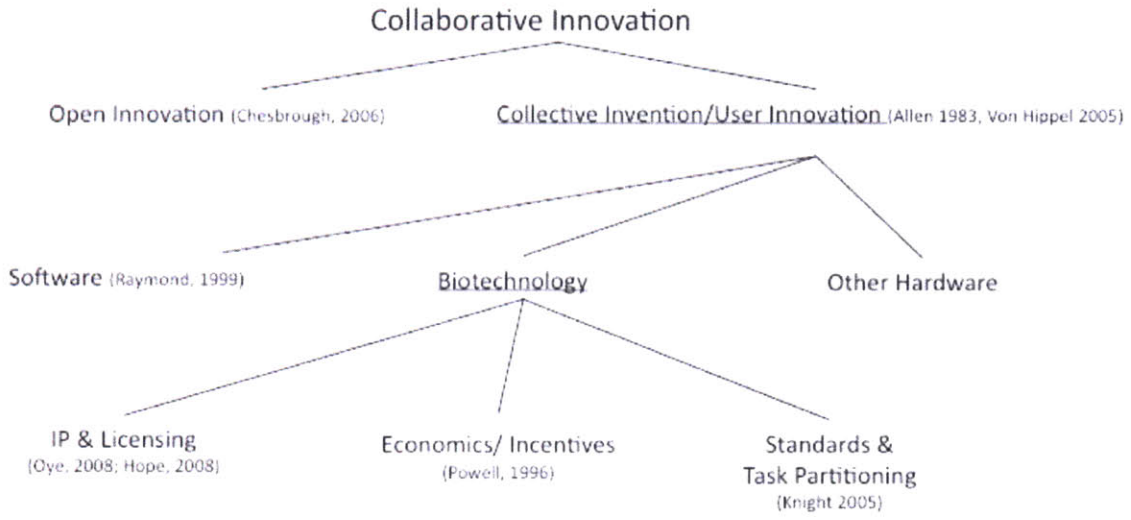


Figure 10: Decomposing the literature on open system design and open innovation.

2.2 User Innovation, Open Innovation, Collaborative Development

Chapter 1 introduced basic distinctions and definitions in the literature on open system design and open innovation and noted that this thesis takes the perspective of collaborative design of complex systems. However, important work has been undertaken outside of this perspective. Figure 10 breaks down the research in open design as a function of technology, with example/important papers listed next to each category, and emphasizes that User Innovation (or Collective Invention) or Open Innovation is not mutually exclusive to it. These three literatures – Collective Invention, Open Innovation, and Collaborative Development - are therefore summarized and compared below.

Collective invention and user innovation is generally used to refer to situations in which design information is mostly or completely free of intellectual property restrictions. In these contexts, users and developers share information about their inventions freely. The notion that *collective invention*, defined by the free revealing of design information, might constitute a distinct method of innovation which originated in a 1983 paper by economist Robert Allen (Allen 1983). In it he described the collective invention of blast furnaces in the 19th century British iron industry. In this case, the sponsoring firms were users of the technology. Over the past few decades Eric Von Hippel, together with an increasing number of user-innovation researchers, have empirically verified a form of technological development and diffusion in which individuals or firms voluntarily relinquish and share inventions (Von Hippel 2005). Some authors claim that "open source software," created through the voluntary contributions of thousands of software programmers around the world, is best conceptualized as a subset of collective invention or user innovation (Osterloh and Rota 2007).

In contrast to collective invention, the term "open innovation" has been used recently to define a more traditional approach to collaborative technological development. This term gained popularity in a recent book by management theorist Henry Chesbrough. The book described that an increasing use of "external and internal knowledge flows" in the innovation processes of large firms was due in part to the realization that most good ideas are outside the boundaries of the firm (Chesbrough, Vanhaverbeke et al. 2006). Open Innovation is, therefore used, to describe a novel but still more traditional approach to technological development. Where collective invention describes the free revealing of design information, open innovation focuses almost solely on the potential for large corporations to develop new products or new paths to markets through business models that emphasize in or out licensing of intellectual property (Chesbrough, Vanhaverbeke et al. 2006).

Still a third set of ideas involving "collaborative systems" and "open architectures" has been developed within the intellectual heritage of academic engineering.

Researchers in the field of systems engineering and systems architecture have long identified collaborative design and development as an important paradigm in the development of complex systems. Scholars of *system architecture* such as Maier and Rechtin have noted that some very large, complex systems such as the Internet can only be developed through collaborative design processes (Maier and Rechtin 2002). Similarly, some researchers argue that as the complexity of underlying technology increases, systems engineering will be best carried out by networks of collaborating firms, loosely coordinated by open standards (Brandenburger and Nalebuff 1996; Sanchez and Mahoney 1996; Norman and Kura 2004).

As Pisano notes, however, it is important to clarify that collaborative systems development may or may not be open (Pisano and Verganti 2008). Organizations such as the department of defense have emphasized "open architecture" strategies which encourage the use of public interface standards and the re-use of off-the shelf components as a way of decreasing development and operations costs (DOD 2003). Also, numerous engineering efforts claim to incorporate such concepts into domain-specific designs (Resnick, Iacovou et al. 1994; Fujita and Kageyama 1997; Schofield and Wright 1998).

We can, therefore, summarize these three literatures as follows: (i) collective invention and user innovation generally refer to the free exchange of design information; (ii) open innovation studies focus on new business models and capabilities associated with sharing knowledge and licensing patents; and (iii) open architectures refer to shared interfaces and re-use of sub-systems. These differing perspectives emphasize the need to precisely define what is meant by 'opening design' within a given research inquiry. Inevitably, these concepts overlap. Notably, some of the differences in definitions stem in part from the disciplinary lenses through which they are viewed and the goals of the researchers.

2.3 Open Source Software as a Concrete Example of Open Design

All three of these literatures encompass the basic process of open source software. Most academic research into open design processes has focused on open source software. What exactly is open source software? Paradigmatic examples include GNU/Linux, a PC operating system and Apache, a server operating system. The defining feature of these programs is that their source code – their basic design – is freely available for anyone to read, change, and improve. Distribution and use of the programs is often governed by licenses that ensure the source code remains open. However, in some cases the code is distributed without such a license.

As a product, open source software is distinguished by the fact that its basic design is not proprietary or patented in the normal sense. However, the importance of open source has less to do with the product itself than the method by which it is created. In The Success of Open Source, Political Economist Steven Weber makes this point clearly:

> "The essence of open source is not software. It is the process by which software is created. Think of the software itself as an artifact of the production process. And artifacts are not the appropriate focus of a broader explanation. If I were writing this book in 1925 and the title was *The Secret of Ford*, I would focus on the factory and assembly line and the organization of production around it, not about the cars Ford produced...Toyota, for example, pioneered lean production in a factory that made cars. Twenty years later, this way of making things had spread throughout the industrial economy." (Weber 2004)

Weber thus asserts that the main innovation associated with open source is the nature of the development process – the unique relationship between the product, legal mechanisms governing design and use, and the broader design community which encourage its production. As Weber notes, open source software should therefore be considered a *development methodology*, in much the same way that, the *waterfall method* or *spiral development* are methods of software development (Figure 11).

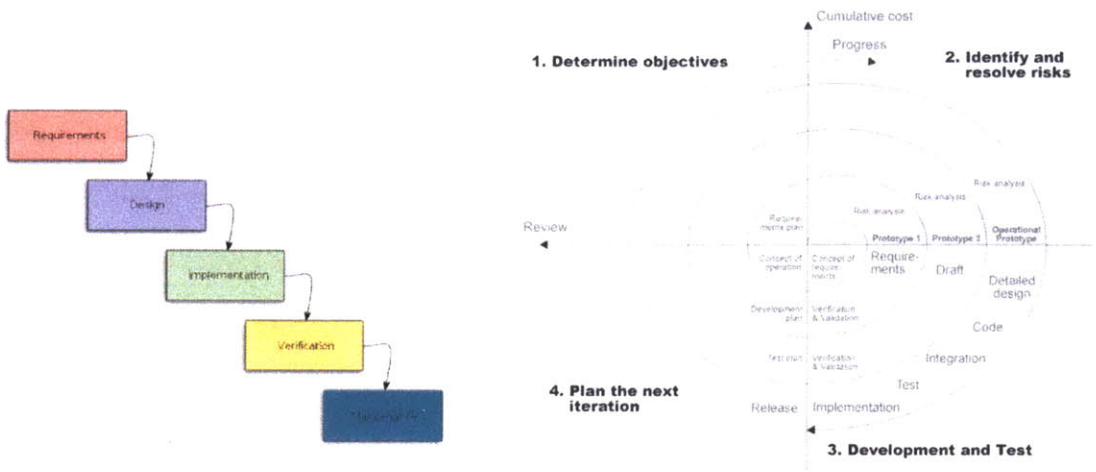


Figure 11: Typical schematics for "waterfall development" (left) and "spiral development" (right). (Source: www.wikipedia.org)

As a development methodology, open source is not fixed. Rather, it can be adapted to the specific needs of the technology and organization employing it. Further, the organizing principles behind open source need not be restricted to software. The guiding principles of openness and distributed design can be used to create a vast range of information-based and even physical products. Whether this is advisable and economically sustainable is a different question.

If open source is a process, what does it look like? Who does the design work and why? How is design work structured? What can be done with the final product? These questions correspond respectively to the structure of the design community, the structure of the design problem, and the nature of the legal mechanisms governing use, distribution and sales. Though many of these questions remain unsolved, empirical research has filled in some of the gaps.

Who does the work? In open source software anyone is free to do the work and designers self-select to problems to which they would like to contribute. Yet, in almost all of the successful examples to date, this occurs after a system architect has created the contours of the effort – both the end-goal and the architecture of the program (Raymond 1999). Further, the effort is not a free-for-all. In almost all practical examples, though the problem is fully open, the design community takes on

a distinct structure. This includes one or two architect(s), a limited number of core coders, and a large number of beta testers and debuggers (Weber 2004). While anyone can contribute code, often one person (the architect) ultimately decides whether to include it in the design for re-release. This structure almost always follows a power law with respect to the number of contributors and the number of contributions (Shirky 2008).

How is the design process structured? With respect to structuring the design problem, open design is therefore related to distributed design processes in general. Because the design problem is intended to be solved and updated by a large number of contributors and users, the design task structure is often explicitly *modular*. This is not always the case, particular in some hardware examples discussed in Chapter 3. However, if a program or design will likely be improved upon continuously by a wide variety of contributors, modularity in design is of great help. To be clear however, this refers specifically to a modularity of design tasks rather than form or function, per se. This is an important distinction – as Von Hippel notes, modular design tasks do not necessarily need to follow modularity of form or of function, though they often do (Von Hippel 1990).

What norms or laws govern distribution and use? The design community and design problem will be bound by a set of legal mechanisms or norms. At the simplest level, there may be a tacit understanding within the community that designs will be freely revealed and intellectual property will not be claimed by designers (Von Hippel 2005). In the open source software community there are now a variety of license agreements with varying levels of restriction. The GNU license originally developed by Richard Stallman includes what is a known as a strong copyleft clause. This requires that all subsequent code must be freely distributed and will be governed by the same permissive GNU license. Other license agreements have been developed that allow users to use source code and put restrictions on subsequent use of their inventions.

The remainder of this literature review summarizes findings on aspects of the basic process of open design. However, due to the prominence of open source software, much of the research was conducted in this domain.

2.4 Open and Collaborative Development in Biotechnology

There is a nascent literature on open source in Biological Engineering. The majority of this work lies in the area of intellectual property (IP) law. A recent Ph.D. thesis from the Australian National University focused on the legal and IP aspects of open source biotechnology, and was subsequently turned into a book (Hope 2004). Arti Rai and James Boyle have recently described how synthetic biology in particular has the potential to be plagued by IP problems associated with information technology (Rai and Boyle 2007). This includes the ambiguity between copyright and patenting, the potential for overly broad "blocking patents," and the converse possibility of multiple, overlapping claims creating "patent-thickets." Rai and Boyle note that a strict open source analogy in synthetic biology would make DNA analogues to source code, which can be copyrighted, but several obstacles stand in the way of this possibility. Work being carried out at MIT's Program on Emerging Technologies has investigated the political economic aspects of IP in synthetic biology (Oye and Wellhausen 2009). More generally, a number of authors question the decision to allow patenting genes which are the product of obvious and widespread research methods. See, for example, (Fellmeth 2005).

Some authors have recently examined the economics of open source regimes within synthetic biology, focusing on the role of network effects in the value of re-usable parts (Henkel and Maurer 2007). In this vein, a recent article examines re-use in the MIT Registry of Biological Parts, proposing some simple organizational guidelines for future registry design (Peccoud, Blauvelt et al. 2008).

While the specific area of open source is not thoroughly investigated in bioengineering, it is worth noting that there is a fairly rich literature on

collaborative innovation in biotechnology more broadly. This literature falls within broader economic inquiries into knowledge networks and the geography of innovation (Ranjay 1998) and collaborative development (Miles, Miles et al. 2005). In Weijan et al., for example, the authors investigate the relationship between innovation (measured as patents) and a number of collaborative ties. (Weijan Shan 1994) In a very widely cited article, Powell et al. examines the dynamics of partnering, concluding that in an industry such as biotech with a complex and rapidly changing knowledge base, the locus of innovation must be considered to be a network not a single company (Powell, Koput et al. 1996). These studies have lead to a flurry of investigations into the causes and implications of collaborative research and design (R&D) in biotech as in, for example, (Feldman 2000; Feldman 2001; Gertler and Levitte 2003).

2.5 Incentives and Micro-Economics in Open Design

A particularly interesting question for economists and political scientists involves incentives for actors in open innovation regimes. Why would an individual or firm spend time developing a technology they do not own? Why would a firm ever participate in an open innovation regime like that documented in Allen's 1983 paper? And once started, why would anyone continue to contribute? These are important questions without obvious answers. Even the architect of the GNU/Linux Kernel, Linus Torvalds, has expressed surprise at the number of contributors to his projects (Rossi 2004).

Social Science researchers have performed a significant amount of empirical work dedicated to understanding individual incentives in open regimes. A good summary of this empirical literature on the motivations for the development of open source software can be found at (Rossi 2004) and (Lerner and Tirole 2005). Lakhani and Wolf, for example, performed a systematic study of 684 open source programmers. They found that a number of non-monetary rewards were important, with "Creative expression" ranking highest, above "user need," "intellectual stimulation," and

"learning" (Lakhani and Wolf 2003). Raymond similarly observed anecdotally that hackers maximize non-monetary considerations such as ego gratification (Raymond 1999).

Beyond software, in a number of papers Von Hippel has laid out the various reasons for which the free revealing of user innovation might be beneficial (Von Hippel and Georg von 2003; Von Hippel 2005). This includes non-monetary rewards such as reputational affects, and those identified by Lakhani. But it also includes monetary and direct benefits such as the sale of complementary goods, or the opportunity for manufacturers to improve products, depending on the nature of the regime (Von Hippel 2005). Von Hippel also stresses that the costs of freely revealing information may be lower than imagined. Many innovations are not patentable, patents are slow, and many ideas are very similar to those already employed by competitors (Von Hippel 2005). With lower costs of disclosure, lower benefits are needed to justify disclosure of inventions.

These observations have lead to a number of formal economic models. In 2001 Kuan developed one of the first formal economic models of user incentives for purchasing off-the-shelf software versus developing their own software (Kuan 2001). Kuan creates a simple closed and open model of software development. In the closed model consumers are divided into "low paying" and "high paying" consumers, whereas in the open model consumers are divided based on their willingness to take part in the design process. Kuan concludes that as the number of consumers willing to act as producers increases, the quality of the software should increase. This is due to the information asymmetry between the seller and the user.

Lerner and Tirole have investigated benefits, costs, and incentives in open source software and have developed a well cited simple model of personal motivations (Lerner and Tirole 2002). They stress that code modularity, "fun problems" and credible leadership (what might be called a respected architect) all contribute to the success of projects. Besson develops a formal model based in part on these kinds of

observations. Like Kuan, Bessen focuses on the aggregate benefit provided by letting users and firms tailor open source software products to their own needs. Based on his model, Bessen concludes that open source development can complement rather than compete with proprietary software, particularly where the product is complex or user needs are highly varied (Bessen 2005).

Von Hippel has also examined the incentives involved in user innovation from a micro economic perspective. Baldwin and Von Hippel examined the decision by users to create their own products from a Coasian transaction cost perspective (Von Hippel 2005). Building on Ronald Coase's insight regarding the organization of markets versus firms, they assume that using the market to buy goods includes a transaction cost. If the cost is high enough, users might decide to make, rather than to buy, the product. Similarly, Krohg and Von Hippel develop a game-theoretic model involving the added benefit to users of improved manufacturing (Von Hippel and Georg von 2003). By disclosing user innovations, the users receive a benefit of improved performance when manufacturers incorporate the innovations.

2.5.1 Conclusion: Incentives and Micro-Economics

There has been an explosion of empirical and theoretical work on the individual incentives of developers in open source regimes. This is especially the case where those developers are also users of the technology. This section reviewed empirical and theoretical work. The range of motivating incentives includes both extrinsic factors such as monetary gain through the sale of services and complementary goods, or practical needs, to more intrinsic factors such as status and creative expression. Recently, Boudreau and Lakhani have summarized this body of literature very well by placing the differing incentives along this spectrum, noting that more than one incentive can be active in a given development effort (Boudreau and Lakhani 2009). Figure 12 below, copied from their paper, places these incentives loosely along this spectrum.

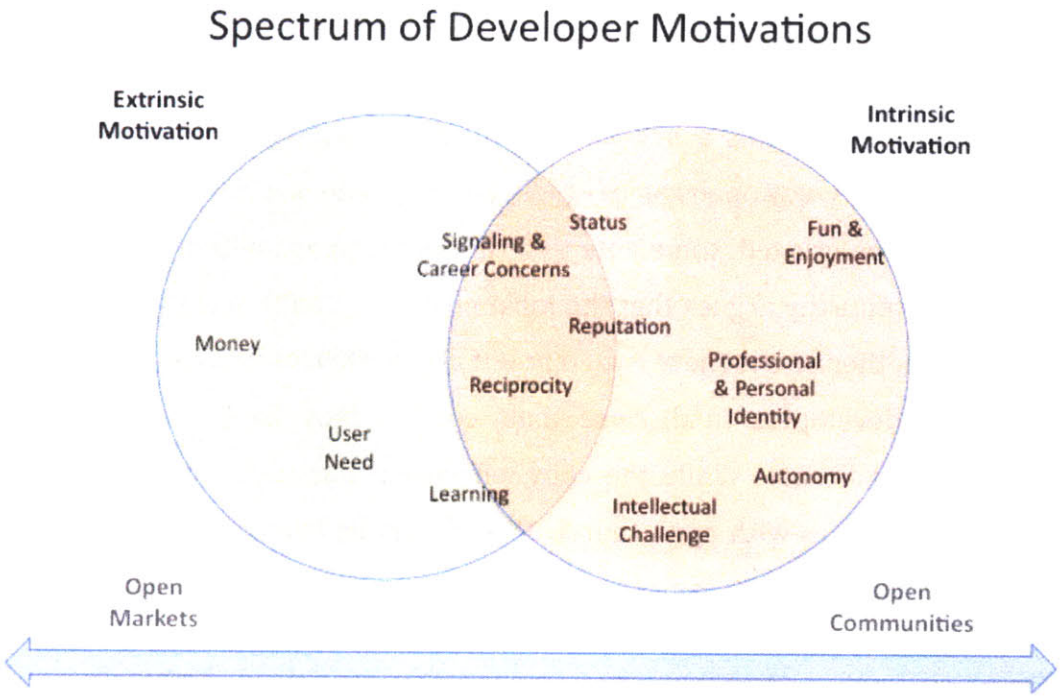


Figure 12: Spectrum of motivations for third-party developers in an open regime. Taken from Boudreu and Lakhani, 2009).

In the framework presented below, the high-level distinction between intrinsic and extrinsic is therefore used, with the understanding that the binary notation includes a range of incentives within it.

2.6 Legal Options and Intellectual Property in Open Design

The intellectual property regime governing an open development process has direct ramifications for the kinds of incentives that will predominate. The first chapter of this thesis outlined a range of legal mechanisms that can govern the use and distribution of technological information within an open regime. Much of the literature on "free revealing" and "user innovation" assumes simply that technological information is given away and placed in the public domain (Allen 1983; Von Hippel 2005). Advocates of open innovation, on the other hand, examine revenue models based largely on licensing patents and other forms of intellectual property (Chesbrough 2003). Between these two poles – public domain and individual patents – lies a vast spectrum of legal mechanisms to protect and pool inventions depending on the goals, organizations, and kind of technology.

A number of authors have examined intellectual property regimes, including the basic categories of patents, patent pools, licenses, and public domain. The Open Source Initiative maintains a list of all available licenses governing open source projects at http://www.opensource.org/licenses. While a substantial number of licenses have been created, some note that the basic options within the licenses are limited. Krishnamurthy argues that the most critical distinction within open source licensing is whether or not there is a copy-left clause (Krishnamurthy 2005). Such a clause forces developers to disclose their code if they have based it on code governed by the license. While the copy-left clause theoretically means that any work which combines with open source code should be revealed, this is not always the case. The Linux community, for example, allows third party developers to write closed code that is compatible with Linux (Torvalds 1999). The range of intellectual property options will therefore be synthesized after analysis of multiple regimes in Chapter 3.

This thesis treats the "legal regime" as a design variable or a parameter which the architect of an OCSD can select, or work around, respectively. For example, GNU/Linux uses the GNU-license, yet many closed-source applications are now sold on top of GNU/Linux. Chapter 3 examines the legal environment surrounding 14 mini-case studies of open innovation.

2.7 Complexity, Uncertainty and the Organization of Design Activity

A number of authors and practitioners have examined the relationship between open design and complexity. These studies often focus on the ability of open regimes to deal with uncertainty, or the possibly organizational advantages that can accrue when developing complex technologies. Before addressing these findings specifically, it is worth emphasizing that many authors assert that these latter questions – and not necessarily incentives – are the most critical questions in open innovation. For example, Jochai Benkler claims that the fundamental benefits and

limitations of collaborative production relate not to incentives, but to the organization of designers and technology (Benkler 2002):

"The incentives problem is trivial if a sufficient number of individual contributors can be networked, and their various sized contributions (each determined by the motivation factors driving any given contributor) can be integrated into a finished product. Modularity, granularity of components, and the difficulty/cost of integration become the efficient limit on peer production, not incentives for contribution or the property rights available in the output." (Benkler 2002)

Similarly, by analogy Weber points out that coherent creation and integration is a much tougher problem than developer motivation:

"The reason a great poem is written by a single person and not by thousands of contributors from all over the world is not that it would be hard to get those thousands of people to contribute words to the collective poem, but that those words would not add up to anything meaningful." (Weber 2004)

Benkler and Weber note that some mix of incentives will often motivate individuals to contribute to a networked innovation community.[6] For these and other researchers, the central questions surrounding open development processes are organizational.

Many authors emphasize that opening can be most useful where sponsors lack information. Opening a partial design or concept to third party developers and users can greatly increase the utility and drop the cost of development when there is significant uncertainty or limited knowledge about the system being developed, the environment in which it will operate, or the needs it will address. In fact, if technological information is inaccessible or rapidly changing, opening may be the only way to gain access to important knowledge.

[6] While these assertions do appear to be supported in the empirical literature cited above it is not clear, as discussed below, whether incentives can be treated completely independent of technological architecture and the cost of integration.

The need for outside ideas can stem from lack of knowledge about the system or from a lack of end-use knowledge. With respect to the latter, Von Hippel argues in Democratizing Innovation that heterogeneity of user needs is often higher than most corporations assume. This is because manufacturers are accustomed to finding user similarity in order to increase volumes, rather than to look for user heterogeneity (Von Hippel 2005). Von Hippel notes that where users have highly heterogeneous needs, companies can profit by letting them customize technology for their own needs. From a different perspective, Chesbrough stresses that companies like Procter and Gamble are increasingly seeking third party collaborators because they realize that "the majority of good ideas reside outside their boundaries" (Chesbrough 2003). Knowledge about both markets and technologies can be gleaned from third party collaboration.

With respect to former, many stress that a lack of system knowledge can stem from the complexity of the underlying design whose development is being planned. For example, in the Mythical Man-Month, Frederick Brooks stresses that the *essential complexity* of software (as opposed to accidental complexity arising from the development effort) creates serious development problems (Brooks 1995). Essentially complexity arises from the internal complexity of the code itself (millions of lines at times) as well as – importantly – the inability to conceive of all the possible operating environments.

One way, of course, to deal with this complexity is to break problems down into smaller, modular pieces. These are easier to solve, but this practice adds the burden of communication between the individuals and the teams that are creating each piece. Assuming that each developer works on one piece and must communicate with every other piece, the number of communication paths will rise with the square of the number of people, resulting in a quagmire of conversations. Building on this observation, Brooks' famously coined *Brooks' Law*: "Adding manpower to a late software project makes it later." This is due to the fact that more developers

create a quadratically increasing burden on communication and information transfer. (Brooks 1995)

Brooks' Law is a great simplification, but it does frame the problems with communication and coordination in the development of complex technologies. Superficially, one would assume that opening the design of a software project would only clutter communications paths and make the problem more serious. Yet, somewhat paradoxically, a number of authors stress that open development processes have advantages over closed processes precisely in their ability to reduce coordination and communication overhead in complex design problems. For example, Benkler argues that open regimes might in some circumstances out-compete closed regimes by more efficiently allocating labor to problems (Benkler 2002). That is, significant overhead involved with coordinating personnel and assigning work is removed when developers self-select to design problems. (Benkler 2002). Eric Raymond makes a similar point through observation of his own open source efforts. In the Cathedral and the Bazaar, he writes:

"The Brooks' Law analysis (and the resulting fear of large numbers in development groups) rests on a hidden assumption: that the communications structure of the project is necessarily a complete graph that everybody talks to everybody else. But on open source projects, the halo developers work on what are in effect separable parallel subtasks and interact with each other very little; code changes and bug reports stream through the core group, and only withinwithin [sic] that small core group do we pay the full Brooksian overhead." (Raymond 1999)

Raymond further asserts that open source regimes can outcompete closed production by more quickly identifying bugs. He famously asserted, for example, that "With enough eyeballs all bugs are shallow." (Raymond 1999) For Raymond, then, it is the natural structure of an open regime – a long tail of small contributors aiding a small coterie of architects – that enables it to overcome Brooks' Law. Important questions arising from this assertion are addressed below.[7]

[7] Traditional managers may here question how an architect can be sure that developers will actually decide to work on the breadth of problems that need to be solved? This is, of course, the incentives

Building on analyses like those of Raymond and Benkler, Weber has written an important book focusing on the organization and governance of communities in open source projects (Weber 2004). Weber scrutinizes the way in which code changes were created and accepted in various open source communities. He concludes that it appears true empirically that open source communities can create professional grade software, but that the number of developers alone cannot explain this. Instead, it is the radical decentralization of the community that seems to play a role. He writes: "[open source] demonstrates the viability of a massively distributed innovation system that stretches the boundaries of conventional notions about limits to the division of labor." (Weber 2004)

It should be noted that these conclusions by Benkler, Raymond, Weber, and others are not universally accepted. A number of practitioners question the basic viability of software created through open processes. Connell, for example, has argued that open source should not be confused with a "Bazaar" since a lead developer is needed to manage the project (Connell 2000). This makes the development effort similar to traditional software development, with an architect and a manager coordinating the activity of numerous coders.

Connell's essential point raises important questions. If, as the empirical literature demonstrates and practitioners admit, open source communities are characterized by a small group of core coders and architects supported by a diverse array of debuggers and testers, how exactly does this differ organizationally from traditional software development? At a certain level, it may simply be a matter of degree. The low cost of communication enables a more radically distributed development model in which tasks are partitioned according to size rather than constrained by

---

question addressed above. As noted, Benkler argues that if the developer pool is large enough and that the problems "modular" enough, the incentives problem solves itself and becomes trivial Benkler, Y. (2002). "Coase's Penguin, or, Linux and "The Nature of the Firm"." The Yale Law Journal **112**(3): 369-446.. With one billion people on the Internet, if a problem is small enough, someone will likely solve it. While difficult to fathom, this argument is based in empirical evidence. The question is, of course, how does this differ outside of software?

manpower. This alone, of course, does not necessitate an open model, but perhaps it enables one. As the work required for each task drops, the incentives required to maintain a developer base will also drop.

More generally, there are some indications in the literature that an open development model has advantages because it creates constraints that match the requirements for successful coding. For example, Frederick Brooks has noted that in most development projects testing requires significantly more time than budgeted. He estimates that a typical project requires 1/3 time for planning, 1/6 for coding, and then ½ for component testing and system testing (Brooks 1995). The failure to budget a sufficient amount of time for testing produces "buggy" code and missed milestones. Yet, by necessity open source projects must be assembled and re-released very often by the core coder group. Also, as noted above, the structure of the tasks and contributions is such that the "long tail" of semi-engaged developers mostly tests the components or the system as a whole. More generally, open processes require clear segmentation of the integral versus the modular code - code that can be developed independently of other pieces. This is good "architecting" in a closed program. Good architecting is a prerequisite to beginning an open system development process.

There are other ways in which open processes might breed sound programming techniques. Linus Torvalds noted that the distributed nature of the Linux development model continuously created a situation in which: "Managing people and managing code led to the same decision." (Torvalds 1999) Though not the focus of this thesis, it would likely be fruitful to explore this hypothesis.

2.7.1 Conclusion: Uncertainty, Complexity and the Organization of Design Activity

A number of the most exciting questions surrounding open development involve the conditions under which these open processes might out compete closed processes. To answer this question, a number of practitioners and researchers explore the ability of open processes to cope with uncertainty by better processing information

about the *design of the system* or the *user-environment* from third party contributors or users. For the framework, we can therefore add two high- evel distinctions in terms of input constraints that must be further explored through case study research in Chapter 3:

1. Level of knowledge about the system design; and
2. Lack of knowledge about the market or end-used

Within information systems in particular, Benkler notes that individuals can self-select to tasks, thus reducing management overhead. Raymond stresses that a small core can complete integrated tasks leaving an army of developers to complete the separable bits. Thus, despite the analogy of a "bazaar" of developers, both advocates and skeptics of open source software agree that a strong leader and a well thought out systems architecture are essential perquisites to the organization of design activity in both open and closed processes. The possible advantages of open processes based on organization and communication are thus theoretically independent from whether the project is open or closed. They seem to be connected to the radical distribution of coding activity and the way in which people are managed.

These findings and arguments provide an important basis for creating a framework with which to design OCSD. Yet, they also leave notable holes. For example, all of the empirical and theoretical analysis on coordination mechanisms focuses on software and information production, rather than on physical or durable goods. Instead the work on durable goods focuses more on the heterogeneity of user needs and high-level technological factors such as the speed at which technological knowledge changes. Open communities developing complex physical goods have been identified, but their organization has not been analyzed.

More generally, the arguments put forth for the coordination advantages in open communities are not yet fully resolved. Most agree that open source projects include a core group of coders assisted by a radically distributed developer base. However,

it is not clear that this differs substantially from closed development, other than that a larger group can help with mundane tasks. It may be that open processes create constraints on organization and coding which mirror sound programming practices for complex systems. Yet this has not been explored in depth in the literature. More work is needed to elucidate the relationships between opening, organizational forms, and the success of development efforts.

2.8 System Architecture, Integrality and Modularity

All of the authors cited in the previous section emphasize that for complex, assembled technologies, modularity impacts the organization of design and therefore the benefits of opening. However, with exception of one article cited in this section, few authors examine the exact nature of this link. This section briefly reviews the roots of studies into modularity and systems architecture, before summarizing one theoretical study of how modularity can impact incentives in open development regimes.

The literature on modularity is vast. For the present purposes, however, we are only concerned with the way in which a design problem is partitioned into pieces. This may or may not be related to the actual modularity of the product function or the product form. This is known specifically as "design task partitioning" as described in (Von Hippel 1990). Task partitioning in the design of complex systems can be traced back to the Nobel Laureate Herbert Simon, who emphasized the importance of the "nearly decomposable" property of the information structure in a system (Simon 1962). A module, according to Simon, was a set of elements within a larger system affected by each other with high probability, but affected by other elements of the larger system with low probability (Simon 1962).

Christopher Alexander has also contributed to the literature on modularity in design throughout his career. His Ph.D. thesis and early work examined the application of matrix methods to the design of complex systems, emphasizing the nearly

decomposable nature of many design tasks (Alexander 1964). Alexander's later work including the famous books "The Timeless Way of Building" and "A Pattern Language" had considerable impact on both building architecture and object-oriented coding concepts (Alexander 1977; Alexander 1979). In The Timeless Way of Building Alexander asserts that well designed buildings all include a limited set of patterns comprised of a local context/problem and design solution. A collection of these patterns creates a language of design. Alexander's emphasis on pattern languages influenced the development of object-oriented programming and, according to Gamma and Helm, directly inspired the well known book "Design Patterns." (Gamma, Helm et al. 1995)

Each pattern, whether in architecture or code, is a module in a large system. However, it is important to emphasize that Alexander was not arguing that modularity is always beneficial. In fact, there was a relatively active movement towards the creation of modular buildings – such as apartment complexes – in the 1960s and 1970s, which Alexander expressly condemned. He stressed that a pattern cannot be used mechanically, but must be adapted to fit the service of the whole. (Alexander 1979)

An excellent article surveying the literature on modularity in design with reference to ownership rights was written by the organizational theorist Richard Langlois (Langlois 2002). Langlois notes that one of the critical issues associated with splitting up a development project involves reacting to the necessary creation of new information: "The tasks in an innovative development project cannot be partitioned in advance, since knowledge is continually changing. In such a case, the modularization of the system (the development project) has to change continually; moreover, the modularization at any point has to take into account the inevitability of re-modularization as learning takes place." (Langlois 2002) The quote summarizes well the dilemma associated with breaking up a problem before a solution is known, and suggests strongly that a fixed architecture must be chosen before tasks are partitioned.

There are a substantial number of articles on the topic of "open architecture" in the systems engineering literature. As a general rule, the term in this literature refers to the use of standards to decrease the acquisition costs of information technology. For example, the Department of Defense (DOD) has recently implemented a policy that states "A modular, open-systems approach shall be employed, where feasible." (DOD 2003) This directive has the goal of reducing development time, easing upgrades and encouraging re-use of components. To date this has focused on software and information systems.

Utilizing the language of open design these kinds of approaches are based on systems engineering practices associated with standardizing interfaces. There has been some effort across domains to develop specific designs for opening technological architectures. For example, Paul Resnick and colleagues developed a widely cited system for filtering news based on open architecture principals (Resnick, Iacovou et al. 1994); Fujita developed on open architecture for robot entertainment (Fujita and Kageyama 1997); and Schofield explores open architectures for the controllers of machine tools (Schofield and Wright 1998). Additional system-specific papers can be found in the area of radar systems, energy systems, and other domains. As a practical matter, the literature on open systems architectures is highly heterogeneous and not capable of generalization, focusing largely on technology specific examples that employ some aspect of standardization. The exception to this rule is in information systems where efforts are made to formalize open standards, such as those underway at the DOD and in internet working groups (Maier and Rechtin 2002).

Few researchers have attempted to develop theoretical frameworks that relate the architecture of systems to the benefits and costs of opening. One exception is the work by Baldwin and Clark describing the way in which architecture affects coder incentives in an open source design process (Baldwin and Clark 2003). By architecture, the authors are referring to the amount of modularity, noting that "for

a given code base, the size of the minimal system, and the size and number of the modules, hence the time needed to code each part, and the value of the different pieces, are fundamental architectural decisions." (Baldwin and Clark 2003)

Baldwin and Clark model open source developer incentives through "two linked games." The first game involves an exchange of effort between coders, enabled by the *non-rivalry* of software (i.e., its ability to be enjoyed simultaneously by an unlimited number of users). The second game is a prisoner's dilemma caused in part by the cost of communication (Baldwin and Clark 2003).

	Developer 2: Don't Work	Work on A	Work on B
Developer 1: Don't Work	0, 0	.5v, .5(v−c)	.5v, .5(v−c)
Work on A	.5(v−c), .5v	.5(v-c), .5(v-c)	v−.5c, v−.5c
Work on B	.5(v−c), .5v	v−.5c, v−.5c	.5(v-c), .5(v-c)

Figure 13: Payoff Matrix for two-player, two-module game in the development of a code base. (Source: Baldwin and Clark 2003).

Using these game theoretic models, Baldwin and Clark find that modularity increases developers' incentives to contribute, assuming that the cost of communication does not increase inordinately. Their conclusions thus formalize the qualitative findings about modularity and organization presented in Section 4.

2.8.1 Conclusions: Modularity and Systems Architecture

A number of scholars in diverse domains have analyzed task partitioning in the design of complex systems architecture. Beginning with pioneers like Herbert Simon and Christopher Alexander, important parameters and foundational concepts in the field have been further developed. By and large, this set of literature stands independently from work on open system design and user innovation. There are numerous articles in the systems engineering literature on open architectures. Yet, this body of work is fairly heterogeneous, focusing largely on point designs for

creating standards in one domain. Importantly, these design studies in open architecture are not linked to the broader discussions about open source development and open systems design in the economic, political, legal, and computer literature. Baldwin and Clark's work provides a starting point for linking the properties of a system's architecture to the incentives in an open regime. Yet, it is theoretical and does not empirically relate properties of the architecture with observable attributes of the development process. Further, it defines "architecture" simply in terms of the amount of "modularity."

For the framework, then, we can note for now that modularity of the underlying design will likely place constraints on the kind of objectives, outcomes, and incentives that predominate in an open regime. However, casestudy work will be needed to identify whether and how modularity correlates to these other factors. We will add a simple input constraint:

1. Modularity of the underlying design.

2.9 Developer Strategies

GNU/Linux, Apache and the scores of additional examples have definitively demonstrated that commercial grade software can be created through open source processes. The above-cited literature analyzes incentives and legal options that enable this development. Businesses are concerned primarily with profitability. Developing products is costly and time consuming. How can investment be recouped if the primary source of competitive advantage and high-margins – Intellectual Property – is abdicated? Even if the development process is faster, cheaper, or better than a closed process, how can a business make money in an open regime?

A growing set of literature addresses these questions. Krishnamurthy examines how open source software can be incorporated into a business model (Krishnamurthy

2005). Distinguishing between "GPL" and "Non-GPL" models, he identifies four basic ways that software companies have found to use open source:

1. *Distributors*: Repackage and sell existing software, often together with support services. Examples as of 2005 include Red Hat, Calder, and SUSE.
2. *Producers not using the GPL License*: In the absence of a license requiring disclosure of software built on existing open source, software producers can incorporate open source code into new products. Microsoft has used this strategy.
3. *Producers using the GPL License:* If a GPL or related license exists, a producer must disclose the source code along with the product it is selling, if derived from or using open source code.
4. *Third-Party Vendors:* Firms can sell services to improve the use of acquired open software.

As Krishnamurthy notes, at a more general level the models come down to two revenue-generating processes: selling modified/complementary software or selling services on the software.

Moving outside of software, both Allen and Von Hippel identify methods for capturing value given user innovation or collective invention. Von Hippel identifies three methods for manufactures to derive value from user innovation (Von Hippel 2005):

1. ***Manufacture*** user innovations
2. Sell ***kits or platforms*** to aid user design
3. Sell products or services that are ***complementary*** to user inventions.

Von Hippel's general cases – which comprehend physical products and include manufacturing costs – encompass Krishnamurthy's software specific business models. While manufacturing is not an issue for software, Krishnamurthy's examples include selling services and complementary products. The sale of platforms that encourages third party designs is well established in software.

If manufacturers can make money in this way the next question is whether they *should*? Would not margins be lower and should not the manufactures strive to keep

design processes, ideas, and property secret for as long as possible? In response to this Von Hippel states simply that quite often where user/distributed innovation becomes possible, the manufactures will have no choice but to adapt or die (Von Hippel 2005). Those that embrace the new models will offer low cost products that better address customer needs. Nevertheless, Von Hippel also stresses that encouraging user innovation may be beneficial in its own right. In particular, this may occur if user needs are (1) unclear and/or (2) rapidly evolving (Von Hippel 2005).

Allen more rigorously examines the economic conditions in which collective invention could be profitable for a business (Allen 1983). Allen's case involves blast furnaces in Great Britain's Cleveland district, yet his analysis is more generic. He first notes that revealing design information may make economic sense if it improves the firm's reputation (marketing) or simply if keeping secrets is too costly. The latter may occur if consultants are used to provide design or manufacturing expertise.

More formally, Allen examines how collective design may be profitable in its own right. He observes first and foremost that collective design can lead to higher performing products in less time. Referring to his case in particular, Allen identifies three scenarios in which collective design may yield higher profits:

(1) If the process is specific to the manufacturing of an asset also owned by the firm, increased efficiency in processing can yield increased profit;
(2) If the process is preferentially beneficial to the distribution of prices for factor inputs to the production process for a company or group of companies; or
(3) If the demand curve is inelastic and the firm owns input then it may make sense to sponsor opening.

All three of Allen's examples assume that the firm also sells other products, whether inputs to or outputs of the revealed invention. In the first two examples Allen notes that the specificity of the complementary good can play a role in the profitability of

disclosing information. Again, this bears relation to the software industry in which Microsoft encouraged third party developers to create windows specific products that increased the utility of the Windows platform. We will return to the question of complement specificity below.

2.9.1 Conclusion: Developer Strategies

Management theorists and economists have identified strategies for developers to generate revenue within OCSD when some technological knowledge and design information is not proprietary. Chesbrough and colleagues focus on business models involving in sourcing and out sourcing of proprietary IP. Von Hippel and Allen both identify three general ways in which companies can profit from free revealing or user innovation. They also assert that quite often the benefits of patenting may be lower than appreciated, and the costs of keeping knowledge secret may be higher than justifiable. Krishnamurthy has catalogued business models in the open source software environment more specifically and which fit within generic categories identified by Von Hippel for user innovation.

At the most generic level, firms pursue two basic strategies within an OCSD, each with several distinct options:

1. Sell support services
    a. Customization
    b. Installation
    c. Fabrication
    d. Repair
2. Sell complementary products
    a. Design kits
    b. Complementary goods
    c. Factor-inputs
    d. Combined goods

2.10 Literature Survey Conclusion

This thesis takes the perspective that creating an Open Collaborative System Development process is a design problem with the principal objective of creating an environment in which others innovate. From this practical perspective, many

academic distinctions that complicate the domain become less important. For example, one cannot focus on creating the right incentives without considering the legal regime or the way in which the design problem is organized. Similarly, the distinction that some make with respect to open innovation, user innovation, open architectures, or open source, becomes less important. From an empirical and theoretical perspective these focused studies are critically important for elucidating and explaining relevant phenomena. From a design perspective, the results of these studies must be synthesized for practical application.

This literature review therefore covered microeconomics, political economy, intellectual property, systems architecture and systems engineering in an effort to extract the relevant findings made in different disciplines. Each section resulted in a list of variables that can be used in the design of OCSD, or a gap in the literature where such a list should be compiled.

2.11 Initial Framework Development

Figure 14 presents the framework outlined in Chapter 1 with the addition of the major taxonomic categories identified in this chapter.

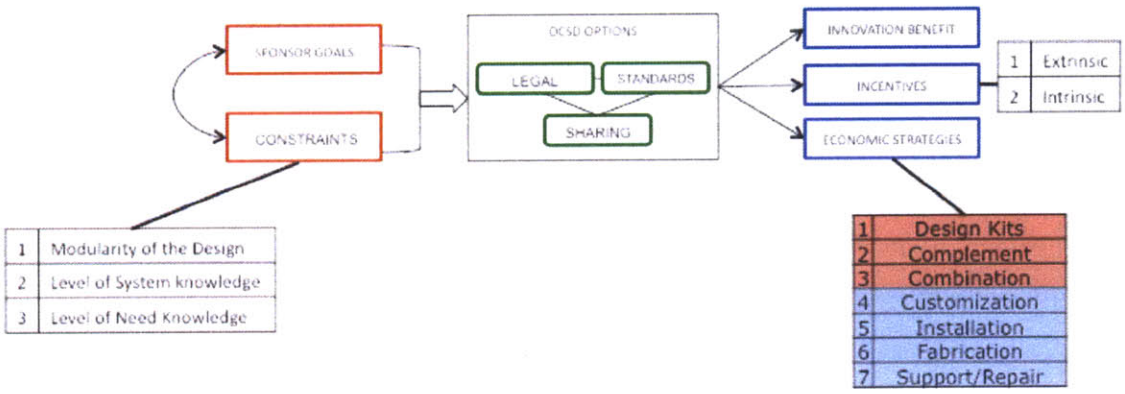


Figure 14: Framework based on results of literature survey. Large boxes represent taxonomic categories: Orange is inputs, green are design variables, and blue are outcomes. Some of the options within each category can be included based on the literature review. For the economic strategies, red connotes products, blue services. Blank areas must be filled out through mini-case studies.

We can assume that the sponsor of an OCSD will have different goals and objectives. These will be affected, in part, by the nature of the constraints associated with the design of the system whose development is being planned. Three of the most

important constraints identified above include lack of system knowledge, lack of knowledge about the end-uses, and the modularity of the underlying design.

A distinction is made between intrinsic and extrinsic incentives, with the understanding that each category has numerous examples. The range of economic strategies can be partitioned into two basic categories: Products (red/top) and Services (blue/bottom). More generally, depending on the goals of the sponsors and the nature of the constraints, one could therefore select from a range of "design variables" including: the IP regime, what is shared, and what is standardized. These design choices will impact the range of economic strategies pursued by third party developers.

This preliminary framework raises three important issues. First, some of the missing dimensions must be defined. That is, while various authors have addressed questions concerning IP, shared information and standards, this has rarely been done in a systematic way across multiple regimes. Chapter 3 develops these "design variables."

Second, as with all design problems, the constraint on choice due to choosing between design variables versus design parameters may be dictated by circumstance or it may be a subjective choice. For example, a sponsor of an open regime may be able to affect the IP regime, but may not be able to chose or design standards. Conversely, the sponsor may have the ability to design standards but have no control over what users and developers share.

3 Thirteen Cases of Open Collaborative System Design

3.1 Introduction

Chapter 2 extracted a taxonomy and a nascent framework from a broad survey of the literature related to Open Collaborative System Development (OSCD). It identified conceptual building blocks from a range of disciplines that can be used to treat OCSD as a design problem, and assembled these blocks into a nascent framework. It also identified some important gaps in our understanding of open development regimes relevant to the framework. Most importantly, beyond high level discussions of modularity, few authors have examined the relationship between the architecture of the system being developed and the various attributes of the open regime.

This chapter examines fourteen previously identified examples of open collaborative system design with the primary goal of addressing these two gaps and further developing the framework. The guiding hypothesis of this case study work is that, besides modularity, elements of the system architecture have an impact on OCSD processes. Because these factors must first be identified and defined through cross case analysis, the chapter employs a pragmatic blend of grounded theory and case study methodology. First, the background is provided for each mini case followed by a brief discussion of the organizational and economic context, the technological context and architecture and the IP regime. The chapter concludes with a cross case analysis of relationships identified through the case study work.

The "mini cases" are based on both primary and secondary source material, spanning a range of domains, including heavy industry, consumer products, biotechnology, aerospace and software. They were selected because previous they fit within the definition of OCSD defined in the introduction. As discussed in the conclusions, both the similarities and the differences between these cases must be considered in order to make valid cross case conclusions.

Table 4: Thirteen mini cases analyzed in Chapter 3. Some of the important cross case observables are listed in this table. Others are listed at the end of each case and in the cross case analysis. Three of these cases are presented in this chapter: Cleveland, Chongqing, and Red Hat GNU/Linux. The remaining cases are presented in Appendix A to the thesis.

	Name	Industry	Technology	Date	Community Barrier	Shared Info	Exchange Medium	Legal
1	Cleveland	Mining Iron	Blast Furnace	1850s	Geographic	Design Performance	Journals / Presentations	Public Domain
2	Bessemer	Mining Steel	Steel Converter	1890s	Patent Pool	Design Performance	Newsletter / Site Visits	Patent Pool
3	Cornwall	Mining Coal	Steam Engine	1800s	Geographic	Design / Performance	Newsletter	Public Domain
4	Chongquing	Consumer Products	Motorcycles	2000	Social Network	Architecture	Social Networks	Public Domain
5	Red Hat	Personal Computers	Operating System	1999	Coding Knowledge	Design	Public Library of Functional Modules	General Public License
6	Google Maps	Web Services	Web Service	2003	Coding Knowledge	API / Developer Tools	Public Library of Functional Modules	License
7	Amazon Storefront	Web Services	Web Store	2003	Coding Knowledge	API / Developer Tools	Public Library of Functional Modules	License
8	Ebay	Web Services	Web Store	2000	Coding Knowledge	API / Developer Tool	Public Library of Functional Modules	License
9	Clickworkers	Aerospace	Database	2001	None	Database	Internet Database	N/A
10	Goldcorp	Mining	Database	1999	None	Database	Internet Database	Contract
11	Alexa	Web Services	Database	2002	None	Database	Internet Database	None
12	Seti	Aerospace	Database	1996	None	Database	Internet Database	N/A
13	SNP	Biotech/ Pharma	Database	2000	Consortium	Database	Public Database	Public Domain

These thirteen cases fall within the definition provided for OCSD in the introduction. To clarify this point, the following table highlights what kinds of components are voluntarily contributed by an unrestricted set of third party contributors.

The chapter concludes by identifying a new set of taxonomic concepts that can be added to the framework. It also identifies important correlations that can guide strategic decision making within the framework. Three basic "strategic bundles" are identified which link sponsor goals to the nature of constraints, the kind of information and standards shared, and the potential outcomes of the OCSD. These additional elements of the framework will then be validated and applied in

subsequent chapters. This chapter presents only three of the fourteen cases. The remaining cases are included in Appendix A to this thesis.

Table 5: How the cases fit the definition of OCSD.

	Case Name	System or Database	Example Components	Some components voluntarily designed by
1	**Cleveland**	Furnaces	Chimneys, Hearths, etc.	Mine Owners
2	**Bessemer**	Steel Converters	Tilting Systems	Railroad Firms
3	**Cornwall**	Steam Engines	Shafts, Etc.	Mine Owners
4	**Chongquing**	Motorcycles	Tires, Engines	Component Companies
5	**Red Hat GNU/Linux**	Operating System	Modules, Kernel, Applications	Users, now Open Source companies
6	**Ebay Developers**	Used Goods Service	Internet Storefronts	Developer Firms
7	**Amazon Developers**	Books Sales Web Service	Internet Storefronts	Developer Firms
8	**Google Maps**	Mapping Web Service	Mashup Sites	Users and Developer Firms
9	**Alexa Database**	Internet Search	Search Websites	Developer Firms
10	**SNP Consortium**	Drug Discovery Database	Datapoints	Pharma Companies
11	**Clickworkers**	Mars Database	Meteor Crater Sitings	The Broader Public
12	**Goldcorp**	Mining Process	Vein Analyses	The Broader Public
13	**SETI**	Radio Database	Computers for Data Reduction	The Broader Public

3.2 Blast Furnaces in Cleveland, England

3.2.1 Background

Allen first articulated the concept of collective invention using a case involving blast furnace development for iron smelting in Britain's Cleveland district between 1850 and 1875 (Allen 1983). In 1850, and with sudden acceleration in 1869, furnace

owners encouraged the publication of detailed design and performance data in trade and academic journals and presentations at professional engineering meetings. The critical design information, according to Allen, was blast furnace temperature and chimney height, both of which had an important impact on fuel efficiency. The result was a steady increase in fuel efficiency of the furnaces, as well as a steady rise in furnace heights and temperatures, leveling off at approximately 80 feet and 1400°F using regenerative heating methods (Allen 1983).

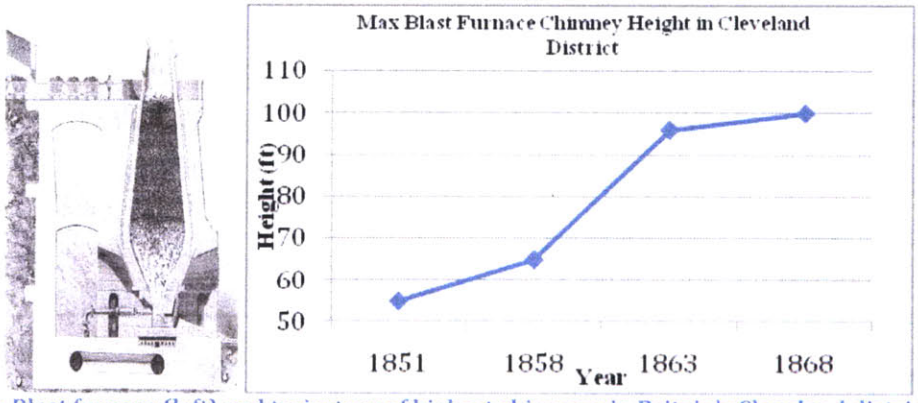


Figure 15: Blast furnace (left) and trajectory of highest chimneys in Britain's Cleveland district (right). Recreated from Allen's data.

3.2.2 *Organizational and Economic Context*

Allen makes a number of observations about the organization of the blast furnace industry that bare on our current analysis. Most importantly, the sponsors of the open regime were blast furnace firms, or users of the technology. The firms also owned or leased iron mines, thus the industry was vertically integrated: "blast furnace firms either owned their own ore mines or leased mining rights at fixed royalties." (Allen 1983) The firms thus owned *complementary assets* that rose in value given any rise in the efficiency of furnaces. A brief calculation by Allen shows that the rise in fuel efficiency in the district exceeded the decline in the value of a given furnace through fuel savings versus the cost of building a new furnace.

Second, the disclosures common to the Cleveland area were bounded loosely by geography. Cleveland competed in the global market for pig iron against rival regions in England, the United States, and the rest of Europe. Given the cost of communication and travel at the time, it would have been difficult for firms outside

of the region to catch up. And even if they did, it was likely that the technical design results were specific to the iron produced in the region (Allen 1983). These two factors – ownership of complementary assets and the specificity of those assets to the opened system – re-appear in the cases below.

Two additional organizational points are relevant. First, the iron market was vast. No change in furnace cost would realistically be transferred to a change in ore prices. Iron companies were thus *price takers*. Second, Allen points out that there was no dedicated research and development (R&D) for mill designs at the time. Thus, as discussed in more detail below, each new design was itself an experiment.

3.2.3 Systems Architecture

The architecture in question involved blast furnace design. This was a highly integral, interdependent, and poorly understood technology at the time. Allen notes that: "In the nineteenth century there was no theory of the blast furnace that would have allowed an engineer to deduce the optimal design from general principles." (Allen 1983) Similarly, "Many aspects of a furnace – its interior lines, the placement of tuyeres, the quality of raw materials, the degree of scaffolding, etc. – exert an elusive but consequential effect on fuel consumption."(Allen 1983) Therefore, in the absence of dedicated R&D, each furnace was essentially a research project leading to an uncertain outcome. Further, it was impossible for the blast furnace firms to break down the problem to each work on individual parts. Instead, the entire design was replicated each time, with minor changes to the two parameters thought to bear most directly on fuel consumption: chimney height and burning temperature.

3.2.4 Intellectual Property

The design in question was replicated and existed in the public domain. The innovations in discussion were incremental. As noted, the main questions for designers involved finding the correct combination of height and temperature. Importantly, according to Allen, these kinds of parameter changes were not themselves patentable. And even if they had been, the rewards for patenting were small. Trade secrets were impractical and thus revealing this information did little

damage, since it would likely leak out anyway. However, sub-elements of the furnaces were patented. For example, in 1865 an engineer patented a design for a firebrick stove that could raise temperatures to 1400°F (Allen 1983).

In conclusion, from an IP perspective furnace owners appear to have had little choice but to reveal design information. Keeping it secret was either costly or nearly impossible. Further, variations in designs were neither sufficient nor practical to resort to protection through legal or trade secrets.

3.2.5 Case Summary

- **Stakeholders**
  - **Sponsors:** Iron Processing Firms
  - **Developers:** Consultants/Furnace Builders and Operators
  - **Users:** Same as Sponsors
- **Objectives & Strategies**
  - **Sponsors:** Lower input costs; Discover better designs
  - **Developers:** Sell Services: construction and repair
- **Technology**
  - **What was shared:** Furnace Design and Performance Data
  - **Modularity of Architecture:** Low
  - **Standards Used:** Measurement standards for comparison only
- **Intellectual Property**
  - **Architecture:** Public Domain
  - **Parts:** Some new parts patented
  - **Standards:** N/A, public measurement standards
- **Constraints**
  - **Value Chain:** Vertically integrated mining and processing
  - **Level of System Knowledge:** Low/No Optimization Models
  - **Level of User-Need Knowledge:** High/Commodity production
  - **Other:** No R&D firms or labs in the industry

3.3 Chongqing Motorcycle Design and Development

3.3.1 Background

Between 1997 and 2002 Honda's worldwide market share of motorcycle sales fell from 90% to 30%. The unprecedented drop has been largely attributed to the rise of new motorcycle design and manufacturing in China, which now exceeds 50% of worldwide sales (Hagel and Brown 2005). Within China the majority of this growth has stemmed from the city of Chongqing. In particular, a number of recent studies have documented how small firms in the city's manufacturing zone design and

develop new motorcycles at a very rapid pace using open, loosely coordinated production networks (Hagel and Brown 2005; Tapscott and Williams 2006). In this model, a lead developer, sometimes a former state owned enterprise (SOE), broadcasts a new motorcycle architecture, specifying design at a very high level including modular elements, module functions, weight, size and interface (Tapscott and Williams 2006). Subsystem designers and producers then carry out the remaining work through relationships mediated through social networks developed at "tea houses and coffee shops" (Brown and Hagel 2005). The resulting bikes are less optimized than a fully integrated design would be, but new designs can be produced more quickly.

3.3.2 Organizational and Economic Context

Some important caveats need to be cited about this case. First, most of the designs are arrived at through the reverse engineering of Hondas and Yamahas. Therefore, in many ways the designs are not new (Hagel and Brown 2005). In fact, the industry was given its start when Japanese firms contracted with State Owned Enterprises (SOE) to manufacture Japanese designs. There appears to be some debate in the literature as to whether the small private ecosystem built around developing and distributing new designs was sponsored by these SOEs or whether the small companies simply used their newly acquired knowledge to develop their own production system. Regardless, each subsystem producer is pursuing a product development strategy.

3.3.3 System Architecture

The motorcycles produced through this reverse engineering process are modular and low performing. However, the modularity of the interfaces still requires communication between subsystem developers. This communication is accomplished, as noted above and according to the literature, through "conversations at tea houses and coffee shops." Therefore, there appears to be a trade off between the ease of development, enabled in part by increased modularity, and the performance of the overall systems. The value is placed on design and development over performance.

3.3.4 Intellectual Property

While intellectual property data was difficult to find given the relatively sparse literature on this case, some assumptions can be made. First, the designs being developed are not patented but, rather, exist within the public domain (or are, perhaps, copied illegally). To the extent that this is common practice in China and given the speed at which parts are manufactured, it is unlikely that subsystem developers are filing patents. Therefore, the intellectual property regime does not provide a great deal of protection for developers. Keeping development costs low ensures returns.

3.3.5 Case Summary

- **Stakeholders**
  - **Sponsors:** Integrating firms/former State Owned Enterprises
  - **Developers:** Local Component Manufactures
  - **Users:** Mass Market
- **Objectives and Strategies**
  - **Sponsors:** Profit/Rapid Design
  - **Developers:** Sell Component Parts within Ecosystem
- **Technology**
  - **What was shared:** Architecture and Standards/Blueprints for similar designs
  - **Modularity of Architecture:** Semi-Integral
  - **Standards Used:** Interface and Measurement
- **Intellectual Property**
  - **Architecture:** Reverse Engineered/Ambiguous IP
  - **Parts:** Proprietary
  - **Standards:** Public Domain
- **Constraints**
  - **Value Chain:** Disintegrated/Decomposed vertically and horizontally
  - **System Knowledge:** Somewhat high/short component dev time
  - **User-Need Knowledge:** Low/Market changes with each iteration

3.4 Red Hat GNU/Linux

3.4.1 Background

Sourceforge.net, a repository for open source projects, currently lists tens of thousands of active projects. This movement has diverse roots, but is perhaps best symbolized by the relative success of the GNU/Linux operating system, and its subsequent commercialization by companies like Red Hat. The latter started with a unique version of GNU/Linux distributed by Marc Ewing in 1994. In 1995 Red Hat merged with ACC Corporation, a company that sold GNU/Linux and Unix software

accessories. Receiving venture capital funding in 1997, the combined company was called Red Hat. It began to make significant advances in 1999 when it struck an alliance with IBM to promote GNU/Linux. That year Dell became the first major company to use Red Hat GNU/Linux in their servers and workstations. Red Hat went public in August 1999. As of late 2009, it has a market capitalization of around five billion dollars on an annual revenue of about $700 million.

3.4.2 Organizational and Economic Context

Like most of the existing pure open source companies, Red Hat replaces a product model with a service or a subscriptions model. Rather than sell a software product, Red Hat integrates hundreds of open source software packages into a stable and upgradable version of GNU/Linux. Customers purchase training, support, and consulting services around this integrated product. The model takes inspiration from commodity industries where products are integrated from often poorly differentiated components. As founder Bob Young writes: "We operate much like a car assembly plant taking parts from many suppliers and building useful products from those parts." (Young 1999)

Of course the "parts" that Red Hat uses are free and available to anyone. Red Hat adds value in three areas. First, by combining products Red Hat creates a conveniently packaged system useful for most customers. Second, it removes reliability concerns by providing consulting and support services. Finally, by revealing source code Red Hat allows customers maximum flexibility to modify the product without concern about breaching licenses. Further, it ensures that the product will be available in the future.

According to a number of scholars, Red Hat solves a number of market barriers to the wider adoption of free software (Weber 2004). These include:

1. **Trust** – A typical mainstream customer will likely hesitate before using a random version of freely available GNU/Linux. Red Hat assures customers that their version has been certified and will work

2. **Reliability** – Similarly, the subscription model assures users that they will have support if problems arise; and
3. **Usability** – Open source products are created "by hackers for hackers." Therefore an often cited brake to adoption includes usability

Once these problems are solved, secondary advantages of the open source model become more apparent. As Bob Young notes, the most unique element of Red Hat's value proposition is the control that users and third party developers gain over the product (Young 1999). This control is guaranteed by Red Hat's use of the GNU Public License.

What then is to stop other companies from selling their product? Young argues that in commodity businesses the main differentiator is often trust which turns into a strong brand. He notes that Heinz has a large market share not because it is highly differentiated, but because consumers have come to equate ketchup with Heinz. (Young 1999)

3.4.3 System Architecture

Red Hat is one of the earliest and most visible companies commercializing a version of GNU/Linux. Much has been written about the code architecture underlying GNU/Linux. Unfortunately, some of it obscures the origins of the operating system. Linux is, technically speaking, a kernel which works within the a broader operating system based largely on the GNU system developed under the leadership of Richard Stallman in the 1980s and 1990s. While the kernel is, arguably, one of the most difficult and important parts of the system, it does not account for the bulk of the code. In 1991, Torvalds wrote the Linux Kernel and, together with an open source community, formulated the last pieces of code and testing that could fashion GNU into a workable operating system. GNU/Linux is thus truly a collaborative product. Created in part in the 1980s, finalized and tested as a complete system in the 1990s, and continuously updated by a community of users and developers since then.

With respect to the architecture of this system, a few high level points are important within the context of this thesis. First, the modularity of the Linux kernel is often cited as a reason that thousands of people can participate in the code. Red Hat GNU/Linux already had over 800 "loosely coupled" program packages in 1999 (Young 1999). As Torvalds notes, "With the Linux kernel it became clear very quickly that we wanted to have a system which is as modular as possible. The open-source development model really requires this, because otherwise you can't easily have people working in parallel." (Torvalds 1999) With the second release of GNU/Linux in 1995 (Linux 2.0) Torvalds included an explicit structure for adding code modules.

Radical modularity is a key element of distributed design. Yet, the technical story does not end there. GNU/Linux was not simply a completely modular collection of code elements. Most of the modularity is in the broader code and not in the kernel, which is a monolithic structure. Torvalds exercised significant control over the basic architecture of the kernel even as he encouraged developers to create modules for the broader OS. He was keen to keep interfaces to the kernel to a minimum, keep the kernel small, and modularize all other code contributions. (Torvalds 1999)

Here Torvalds interestingly observes that these and other design decisions stemmed both from the requirements of the distributed developer environment and the requirements for successful long-term viability of the operating systems. As Torvalds said, in many cases, "Managing people and managing code lead to the same decision." (Torvalds 1999) In this case, design constraints stemmed from the need for continued improvement in the code. Management constraints stemmed from the need for one architect to feasibly manage the contribution of thousands of volunteers while also ensuring that these volunteers did not, "step on each others toes."

There are four identifiable guidelines throughout Torvalds' writing that seem to align the distributed design and management requirements:[8]

1. Keep kernels small;
2. Minimize constraints on future development activities;
3. Delay interface decisions and minimize interfaces to the core kernel; and
4. Modularize code contributions.

This may merit further investigation, though it is outside of the scope of this thesis. Suffice to note that it seems that the constraints and requirements imposed by distribution development are also those imposed by "evolvability." Both are central to the open innovation processes.

3.4.4 Intellectual Property

Red Hat Linux, like all versions of GNU/Linux developed from the GNU operating system, is governed by the GNU Public License (GPL). The basic features of this license were discussed in Chapters 1 and 2. The GPL is generally considered one of the more restrictive open source licenses. It permits anyone to use the code. However, a copyleft clause requires that anyone using the code must distribute, in kind, any source code that builds upon it. This theoretically ensures that users will return their software developments to the public commons.

However, some aspects of this license remain unclear, especially as they pertain to GNU/Linux. As Torvalds notes, the copyleft clause should theoretically prohibit the use of closed source third party software applications. Yet, they exist partially through necessity. Torvalds writes:

"We ended up deciding (or maybe I ended up decreeing)...any program running on top of Linux would not be considered covered by the GPL....Because of this commercial vendors can write programs for Linux without having to worry about the GPL...this is still a gray area of the kernel though. These gray areas leave holes for people to take advantage of things....But I don't think anyone wants to misuse the kernel; those who have shown commercial interest in the kernel have done so

[8] These are extracted from various parts of his writings in (Torvalds, 1999)

because they are interested in the benefits of the development model." (Torvalds 1999)

In other words, while GNU/Linux itself is in theory governed by the GPL, in practice the broader GNU/Linux ecosystem is not. This mixed legal regime runs somewhat contrary to the original spirit guiding the development of GNU. For companies it also creates legal uncertainty. However, as Torvalds notes, it is unlikely that anyone in the community would try to exploit this grey area. Interestingly, the mixed regime makes the case that the GNU/Linux ecosystem and Red Hat in particular, are more similar to some of the other cases described in this chapter. A free, core architecture is supplemented by both free and proprietary modules. Whether or not this was the original intention of Stallman and the developers of GNU, it suggests that at least for now the two regimes can coexist thereby producing substantial benefit for society.

3.4.5 Case Summary

- **Stakeholders**
  - **Sponsors:** Originally, GNU project. Then Linus Torvalds. Now, multiple sponsors.
  - **Developers:** Red-Hat; Other companies; Open Source Community
  - **Users:** Open Source Community; Corporate Customers
- **Objectives & Strategies**
  - **Sponsors:** Low-cost OS development; Fun
  - **Developers:** Sell services; re-configured code versions
- **Technology**
  - **What was shared:** Source code for GNU/Linux Operating System
  - **Modularity of Architecture:** Low in kernel; High elsewhere
  - **Standards Used:** Interface Standards for modules; application interfaces (APIs)
- **Intellectual Property**
  - **Architecture:** GNU Public License
  - **Parts:** Kernel and Some Modules Under GNU; Applications can be closed source
  - **Standards:** GNU Public License
- **Constraints**
  - **Value Chain**: Disintegrated horizontally and vertically; Linux is a layer
  - **System Knowledge:** Relatively High; Low tech-dev risk
  - **User-Need Knowledge:** Low – many different kinds, customization valuable

3.5 Case Summary and High Level Conclusions

This chapter surveyed fourteen cases of open collaborative system development identified by previous authors. Three cases were presented here and the remaining are included in Appendix A to this thesis. Prior to conducting a cross case analysis a number of general observations can be made. First, the cases are highly

heterogeneous with respect to sponsor motivations and developer strategies. Although all of the cases fit within the basic definition of OCSD defined in Chapter 1, the reasons for opening and the constraints faced by firms, organizations and developers differ. For example, the Bessemer steel case resulted from the resolution of a patent dispute, the rise of Red Hat resulted from an academic experiment, and the API strategies from a concerted effort to enable third party developers. To be sure, some of these differences are due to the constraints on the underlying technology and the cost of communication, as discussed below in more depth.

Despite this heterogeneity some general similarities exist. Most importantly, all of the cases involved a complex, integral core, around which third party innovators can build. Depending on the case this core could be a software platform (API strategies and the database strategies), a kernel within an operating system (GNU/Linux), an entire technological design (Cleveland, Bessemer, Cornwall), or a mix of all three (Chongqing). The way in which design tasks are distributed and build upon this core varies across the cases, as analyzed in more detail below, but the basic pattern of integral core and distributed design is consistent.

Second, at a high level, we note that every case also involves a mixture of proprietary and non-proprietary development. While the relative level of non-proprietary information varies – Cleveland furnaces were fully open while Google maps began with an open interface alone – in every case there was a creation of proprietary elements within or upon the free or open element. In all industrial-era cases, for example, patents played a role. In the Cleveland case the technology was free in part due to a lapsed patent. Patents on subcomponents were also common. For the API strategies, the databases are proprietary and "closed source," though some open source is also now used. Even in the case of GNU/Linux - a model for free and open software – third party developers now create closed source complementary applications, which may violate the GPL. The two exceptions to this observation are the NASA Clickworkers project and SETI@Home. However, these projects prove the rule. Both are scientific efforts sponsored by a federal agency

with the sole goal of searching databases efficiently. The sustainable economic examples of OCSD all involve some version of proprietary information.

This observation contradicts what some scholars assume to be the basic phenomenon of OCSD. None of these cases involve armies of unpaid workers creating purely free software or designs, even when those armies provide the impetus to disclose. Instead, the public information serves as a catalyst for a broader ecosystem of non-profit and for-profit activity. One might say that it creates a kind of neutral space, or opening, in which a large variety of activity can take place. Like a public square, which supports a market on some days and public performances on other days, the freely available information creates a platform for activity that might otherwise be too difficult or too costly. Rather than diminishing the importance of OCSD, however, this metaphor suggests that it is all the more important to release the right kinds of information, and to encourage the right kind of competition, to catalyze innovation. The cross-case analysis below delves into these questions in more detail.

3.6 Cross-Case Correlations

3.6.1 Stakeholders and Their Objectives

The cases exhibited diversity of sponsors and sponsor/developer goals, though some patterns do emerge from a cross case analysis. Figure **16** presents a cross case comparison of the sponsors, developers and their objectives and strategies. At the highest level, all stakeholders used OCSD to increase the rate of innovation by reducing the transaction costs associated with acquiring and building on successful designs. Increasing the innovation rate, however, provided different kinds of benefits to different sponsors. These benefits, which are defined in this section as the sponsor's real objective, can be grouped into two distinct categories and correlated to the kind of sponsor. Specifically, sponsors that are users of the technology typically used an increased innovation rate caused by an OCSD to decrease input costs. Sponsors that were developers benefited from increased

innovation rates by access to diverse markets more quickly. These patterns and correlations are described more clearly in the section below.

	Stakeholders			Objectives & Strategies	
	Sponsors	Developers	Users	Sponsors	Developers
Cleveland	Iron Mining Firms	Furnace Builders & Operators	Iron Mining Firms	Lower input costs; Find better designs	Services, Construction, Repair
Bessemer	Steel Firms	Steel Firms	Railroad Firms	Avoid patent thicket	License to railroad firms
Cornwall	Coal Mining Firms	Consultants/ Operators	Coal Mining Firms	Lower Pumping Costs	Sell design and maintenance services
SNP Consortium	Pharma Companies	Universities, IT Firms, Pharam Companies	Pharma Companies, Scientists	Lower input licensing costs, avoid patent thickets	Same as Sponsors
Chongquing	Local Motorcycle Integrator Firms	Local Integratrs & Component Manufacturers	Mass Market	Rapid Development, Time to Market	Sell components to sponsors
Red Hat GNU/Linux	GNU Project & Linus Torvalds	Users, then Red Hat, etc.	Open Source Community. Now Mass Market	Low Cost Development. Free Software.	Creative Expression. Sell services and custom versions.
API Model: Google, Ebay, Amazon	Internet Firms - Ebay, Google, Yahoo	Internet Firms & Third Party Firms	Mass Market	Rapid Development to Widen Market	Leverage Infrastructure for Low-Cost Innovation
Open Databases: Clickworkers, Goldcorp, Seti	For & Non-Profit Orgs; Goldcorp; NASA	Individuals or Firms	Sponsor Firms	Search Databases	Depends on Reward Structure; Fun; Money; Etc.

Figure 16: Cross Case Analysis of Stakeholders and Stakeholder Objectives. Orange signifies sponsor-users. Blue signifies sponsor-developers. Yellow indicates cases in which the sponsor had the goal of lowering input or production costs. Purple indicates cases in which the goal is reducing time to market and expanding markets. Developer goals are highly heterogeneous.

Looking across the cases, a number of patterns emerge with respect to the stakeholders and their objectives. At the highest level, in every case the sponsor of the regime was either the user of the technology (sponsor-user) or the developer of the technology (sponsor-developer). These are indicated in orange and blue respectively in the figure above. The sponsor-users include the Cleveland Case (Iron Mining Firms), The Cornwall Case (Coal Mining Firms), the SNP Consortium (Pharmaceutical Companies), and the Open Database Cases (NASA Clickworkers, Goldcorp Mining). For example, in the Cornish Engine case, coal mining firms created the Lean Engine Reporter as a mechanism for engine designers and operators to exchange design and performance information. These firms were *users* of the steam engine technology as well as *sponsors* of the open regime.

Sponsor-developers include the Internet platform firms under the API model, the Chongqing Motorcycle Development case, and the Bessemer Steel case. For example, in the Chongqing case motorcycle integration firms disclose blueprints and spread design information to encourage rapid, collaborative develop by third party

component firms. Similarly, Internet platform firms such as Google and eBay have proprietary stakes in certain elements of their platforms, but provide code libraries and APIs in order for third parties to develop complementary products.

While most of the cases can be divided neatly into sponsor-users and sponsor-developers, Red Hat GNU/Linux is mixed. Most of the GNU operating system was developed under the supervision of Richard Stallman in the GNU project. Yet Linus Torvalds created the Linux Kernel later. The Linux ecosystem is now supported and, therefore, sponsored in part by for-profit firms like Red Hat GNU/Linux. What started as an effort by users to sponsor an open regime under GNU has morphed into a regime in which for-profit development firms create and disclose open source code. Because the case in this thesis focuses on Red Hat, it has been categorized under sponsor-developers. But the case demonstrates that both users and developers can co-sponsor an OCSD regime. It also suggests that where value has been created through an OCSD, as in the GNU project, broader economic forces can come into play, and these can lock-in the open regime. This, of course, is attributable in part to the nature of the GNU license. But as the detailed case discussion illustrates, the license itself cannot be the only explanation.

The distinction between sponsor-users and sponsor-developers correlates strongly to the sponsor's objectives. For the most part, sponsor-users had the objective of ***reducing input costs***. Sponsor-developers sought to ***decrease the time to market diverse market niches***. These are highlighted in yellow and purple in the figure above.

For example, the Cornish mining firms sponsored an open regime to decrease the cost and increase the performance of steam engines that were used to pump water out of their mines. Steam engine construction and operation was technically a "factor input" cost which could be reduced by sharing design information and best practices. More broadly, however, the exact way in which an OCSD regime reduced input cost varies from case to case. For example, in the Bessemer and the SNP

Consortium cases sponsor-users sought to avoid excessive licensing fees caused by multiple conflicting patent claims. In the Goldcorp and NASA Clickworkers cases, avoiding the need to hire employees reduced input costs. Most of the other cases sought to lower input costs by improving the performance/cost of a factor input.

In the cases in which sponsors of the regime were developers, the objective was usually to increase the diversity of end applications and decrease the time-to-market for these applications. For example, Google publishes the APIs and some source code relevant to its mapping system in order for third party developers to incorporate these codes into a diverse array of end applications such as housingmaps.com. Similarly, Chongqing motorcycle developers are primarily concerned with producing a range of models in as short a time as possible.

One exception to these correlations was the Bessemer Steel case. In this case the open regime was a patent pool and the sponsors were developers of the technology. However, their goal was not to create a diversity of end applications, but rather to remove transaction costs associated with cross licensing the technology. This case, however, was also an exception in that the patent pool was itself not fully open, and it eventually became a closed company. Thus, in a way the case might be an exception which proves the rule for truly open regimes.

Finally, it is important to note that some aspects of the sponsors and their objectives do not correlate across cases. The developer strategies in particular were highly varied, covering the complete range of strategies and motivations articulated in Chapter 2. This is important because it suggests that while an OCSD can be tailored or designed, the specific profit making strategy or motivation for developers may be less important as a design variable. A range of developer strategies may be able to coexist, or to be tailored to the specific needs and resources of the sponsor. However, this hypothesis would need to be verified through further research. At the level of abstraction used to analyze the mini cases, the most that can be said is that

developer strategies do not correlate to the type of sponsors or to the sponsor objectives.

At a broader level it is evident that all sponsors sought to *increase innovation* by *lowering transaction costs* associated with collaboration and design. For some these transaction costs would have resulted from patent thickets (SNP, Bessemer). For others they would have stemmed from negotiation over rewards (Goldcorp) or may have been caused by negotiations over collaboration terms (Bessemer, Cleveland, Cornwall, etc). Increasing the pace of innovation by removing barriers to entry and lowering transactions costs can thus be considered a means by which the sponsors achieved their primary goals of diversifying markets or improving product performance.

3.6.2 Legal Options

A fixed set of high level legal mechanisms govern the ownership and use of technological information across the cases. These mechanisms will not surprise legal scholars since they cover the range of ownership protections available to firms and individuals, as well as some non-traditional licenses developed for open source software communities. Table **6** provides a cross case summary of the legal mechanisms governing the use and distribution of architectural design, parts and standards within each OCDS.

Table 6: Intellectual Property governing the release and use of system architectures, parts and standards across the cases. "PD" stands for public domain. Only interface standards are considered since descriptive standards are covered under "architecture" and measurement standards are always PD.

	Intellectual Property (IP)		
	Architecture	**Parts**	**Interface Standards**
Cleveland	PD	Some patented	N/A
Bessemer	PD	Patent Pool	Patent Pool
Cornwall	PD	Some Patented	N/A
SNP Consortium	PD	PD	PD
Chongquing	Ambiguous - Reverse Engineered Designs	Trade Secret, Not clear on Patents	PD
Red Hat GNU/Linux	GNU Public License	Some under GNU. Some proprietary.	GNU License
API Model: Google, Ebay, Amazon	Company Specific Open Licenses	Some Proprietary, Some Open	PD
Open Databases: Clickworkers, Goldcorp, Seti	Company Specific Open Licenses	N/A	PD

Table 2 illustrates that a range of legal mechanisms can exist with OCSD. We can use this to create a limited list of options:

1. Public Domain (SNP Database, Cleveland Furnaces, Cornish Engines, Chongqing, SETI, Clickworkers, Goldcorp);
2. Copyleft licensing (GNU/Linux);
3. Private Company-Specific API licenses (API Strategies);
4. Patents (All non-IT cases except for Chongqing);
5. Patent Pools (Bessemer); and
6. Traditional, non-IP property rights (All cases with mines).

The list demonstrates that a variety of mechanisms can be used to govern ownership and use within an OCSD regime. This result is a consequence of our definition of an OCSD process. Because we did not limit "open" to mean "open source", "free revealing" or simply "open architectures," the cases could include a range of ownership options almost by definition. However, to the extent that sponsors in these regimes explicitly created or adapted to this range of mechanisms governing all, or part, of the systems being developed, they should all be considered as viable legal options for creating an OCSD.

3.6.3 Knowledge of the System or End-Use

Chapter 2 emphasized that an OCSD process can be employed when sponsors lack information about the technology being developed or the market in which it will be used, whether due to high complexity or rapid change. The cross section of mini cases analyzed here proceeds one step further. It demonstrates that for sustainable open regimes the two dimensions seem to be mutually exclusive and that this kind of uncertainty correlates to the kinds of innovation most prevalent within the regime.

Table 7: The level of system knowledge and user need knowledge across the cases.

	System Knowledge	User-Need Knowledge
Cleveland	Low	High
Bessemer	Low	High
Cornwall	Low	High
SNP Consortium	Low	High
Chongquing	High	Low
Red Hat GNU/Linux	High	Low
API Model: Google, Ebay, Amazon	High	Low
Open Databases: Clickworkers, Goldcorp, Seti	Low	High

Table **7** presents the level of knowledge in the system and the user need for each case. These categories could also be called "system uncertainty" and "environmental uncertainty." A well-defined set of user needs is both stable and homogenous across a market. Enterprises therefore can plan the kinds of products that will fulfill these needs. A market in which customization is required for each new product or new iteration does not have well defined and stable user needs. Similarly, if system knowledge is well defined the governing equations to the technology are known and the technical and development risk is low (though not necessarily low cost). Generally, software projects entail low technical risk due to well-defined system knowledge.

Figure **17** demonstrates that every case contains either poorly defined technological knowledge, or poorly defined user needs, but never both or neither. This finding is consistent with the previous observations that opening can help cope with uncertainty, but it also goes a step further. It suggests first that uncertainty is almost always at the heart of sustainable open regimes and that sustainable open regimes typically deal with only one major category of uncertainty. What kind of technology might exist in the off-diagonals of Figure **17**? These are regimes that involve *both* well defined or *both* poorly defined needs and system knowledge. The former might include stable, commodity industries such as ball bearings or soda cans. The latter –

poorly defined technology and user needs – encompasses basic research and development.[9]

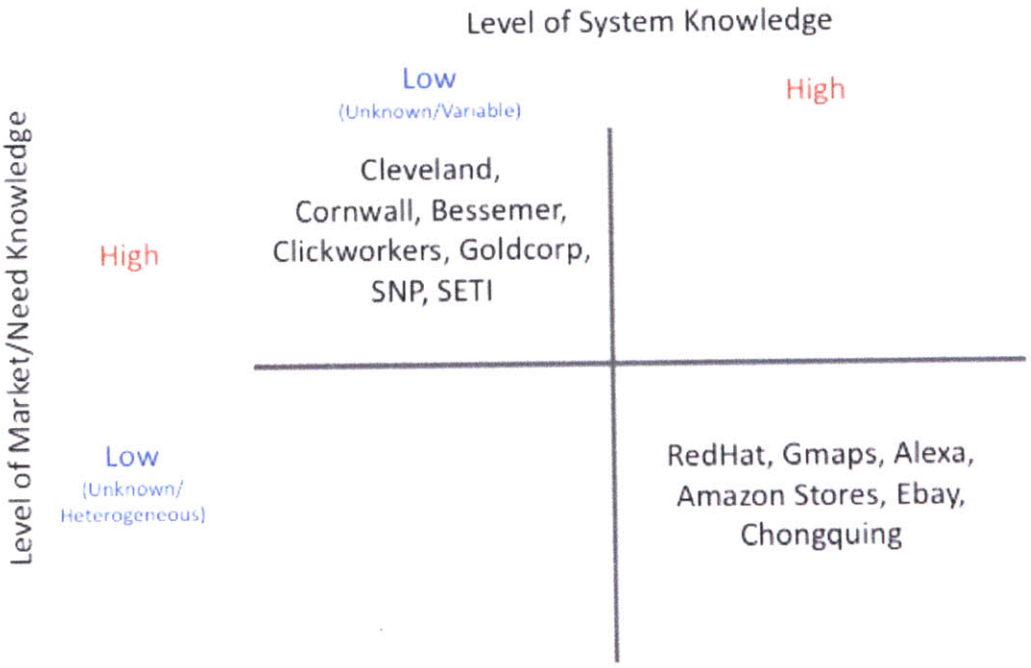


Figure 17: Matrix of needs and system knowledge.

Because each case involves either system uncertainty or market uncertainty, but never high levels of both, we can now correlate the kind of uncertainty present with other aspects of the regime. Figure **18** analyzes how the type of uncertainty relates to the type of innovation most prevalent in the regime. Architectural innovation corresponds to changing the components or the relationship between components in the technology (Henderson and Clark 1990). Incremental innovation is defined as an improvement to one or more components or parameters within the system without changing the architecture.

[9] One might note that norms for sharing data have already been developed and encouraged in the scientific domain. Also, the MIT Registry of Standardized Biological parts falls within this latter category. The end goals for the registry, in terms of user needs beyond basic research and development, are not defined.

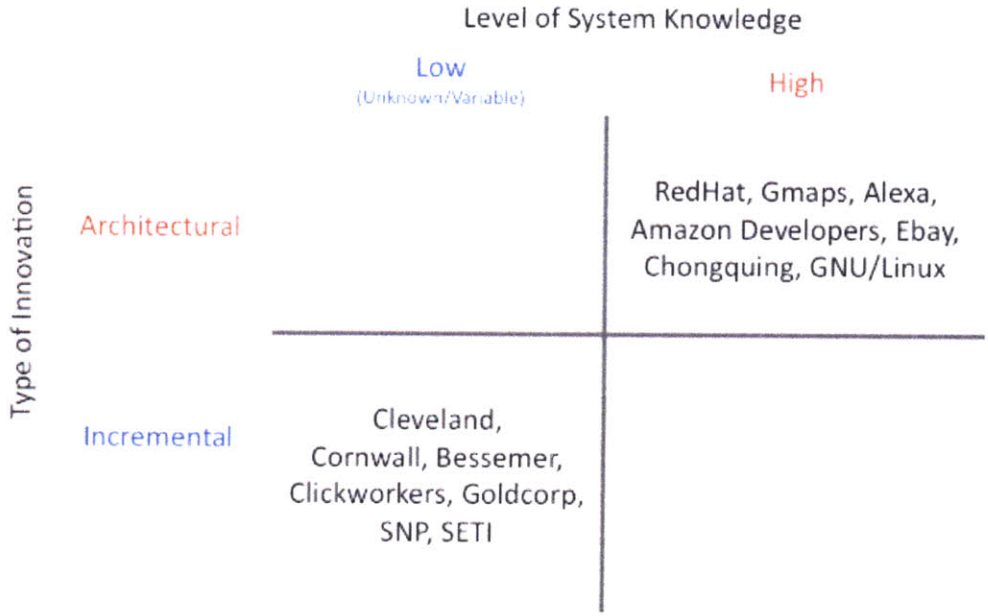


Figure 18: Matrix of uncertainty and type of innovation.

This comparison demonstrates that all of the cases in which system uncertainty is high involve incremental innovation. Also, all of the cases with market uncertainty involve architectural innovation, at least within the part of the design that is open.

A number of possible factors might explain this finding. It could be a simple matter of definitions. It is very difficult to carry out planned architectural innovation if the core technology – and thus the interface between components – is poorly understood. Poor system knowledge thus implies low architectural innovation. Yet, this is not a satisfying conclusion because it is also clear that if the system in not fully understood one would expect to see a great amount of architectural tinkering. Why would an open regime never have developed to share information regarding architectural tinkering?

Therefore, if the gap in the top left box is not an artifact of the case selection, it needs to be explained. There are two possibilities that merit further research. First, sustainable open regimes cannot be developed where system knowledge is low but architectural innovation is being carried out. Second, sustainable open regimes of this nature could be profitably developed, except that too much foresight and

coordination among sponsors, developers and users is required. There is also a gap in the region where user needs are poorly defined and innovation is incremental. It may be that incremental innovations might rarely, if ever, enable access to new markets. They would not, therefore, be useful for clarifying user needs.[10] This should be examined in more detail.

More generally, both of these correlations provide important information on what might create an OCSD, while raising important procedural questions requiring more work.

3.6.4 Architecture, Modularity, Integrality

The technologies analyzed in each case are not all highly modular. Figure **19** bins design of the systems whose development was open into one of four categories: Highly Modular, Loosely Coupled, Semi-Integral, and Integral, using the definition of modularity articulated in the literature review.[11]

[10] An exception to this latter point may involve continuous performance variable such as fuel efficiency in automobiles. Depending on incomes, end uses, climate and other preferences, automobile users will have widely varying thresholds for fuel efficiency. Yet, these various thresholds can be met through incremental changes to the core architecture of the internal combustion engine. We could speculate then, that if incremental innovations are sufficient to meet varying user needs, a company will be better off developing an internal family of products. The incremental cost to the company is low, as opposed to meeting varying user needs through architectural innovation.

[11] Modularity of a database was defined as the extent to which the interpretation or performance of data points or sub-systems is *dependant on other data points* or *subsystems,* respectively.

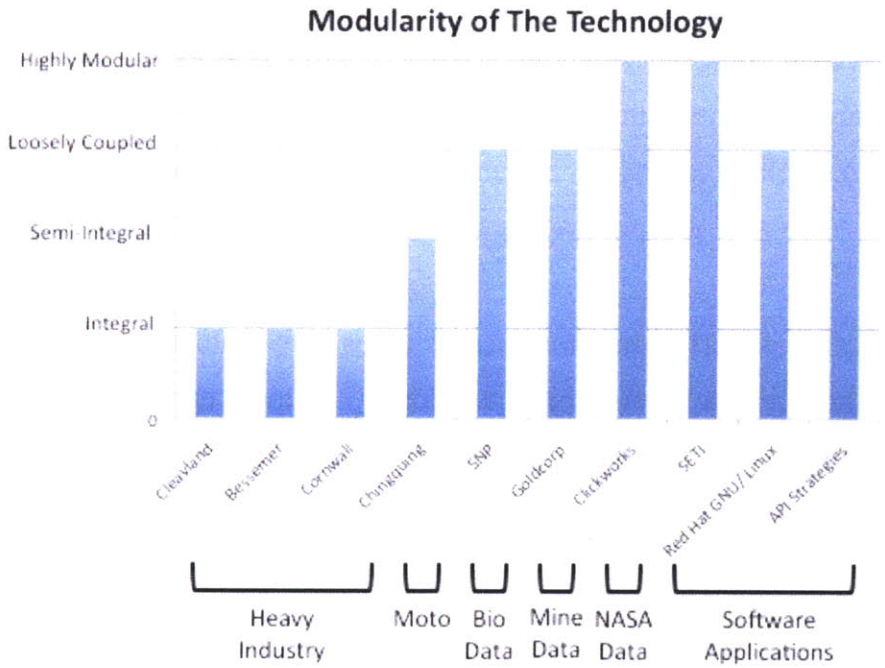


Figure 19: Modularity of the technology in the 13 cases. Three software cases are (combined?) binned together under API strategies.

While there is certainly an element of subjectivity to this measure, it is clear that the form of the technology does not alone determine whether a sustainable open regime can be created around it. That is, it is not *a priori* obvious that opening would not work for an integral, tightly coupled system, as many would expect.

However, in Industrial Revolution cases we find that the basic tasks being performed by individual developers were independent. That is, even though the technology is highly integral/integrated and the entire design freely revealed, developers focused on a limited number of parameters that could be modulated without affecting the rest of the architecture. For Cleveland Furnaces, these parameters were chimney height and blast furnace temperature. For Cornwall, these parameters included cylinder sizes, lengths, and pressures. Therefore, one might view the modularity of the technology slightly differently. In every case an integral core existed around which developers created wholly new systems with small variations. However, in some cases this integral core was proprietary and provided by a company (Google Maps). Where clean decoupling was impossible, this core was freely revealed to all participants.

In short, where integrality would create coordination problems in the cases analyzed here, the entire integral element is freely revealed. This side steps the problem of coordinating design activity by *open and freezing* the majority of the design.

The observation is fairly consistent across cases. For example, Chongqing and Red Hat GNU/Linux involve semi-integral and loosely coupled technologies respectively. The Chongqing case involved the disclosure of the high-level architecture of different motorcycle designs for which developers create proprietary components. GNU/Linux involved the creation of an operating system based on a core kernel and fixed functional elements developed and revealed by Stallman and facilitated by Torvalds. The approach to opening when the technology is integral should be contrasted with the open API and database cases. Sponsors of the latter created interfaces without specifying function of the connecting modules.

3.6.5 Sharing Architecture versus Design Information

A distinction can be made between the kind of information revealed in each case. Beyond the spectrum of how much of an overall architecture is revealed, lies a basic difference between revealing interface information and revealing the entire design. For example, sponsors of Red Hat GNU/Linux and the Industrial-era cases revealed design information for entire systems. This is in contrast to the API strategies, or to the Goldcorp mining case, for which sponsors released interface information while keeping elements or databases fully proprietary and closed. Figure **20** distinguishes between these elements of information sharing. It divides the cases according to whether standard interfaces (both physical and functional) are revealed and whether design information is revealed or provided.[12]

[12] Standards can also include measurement and descriptive standards. These are addressed later.

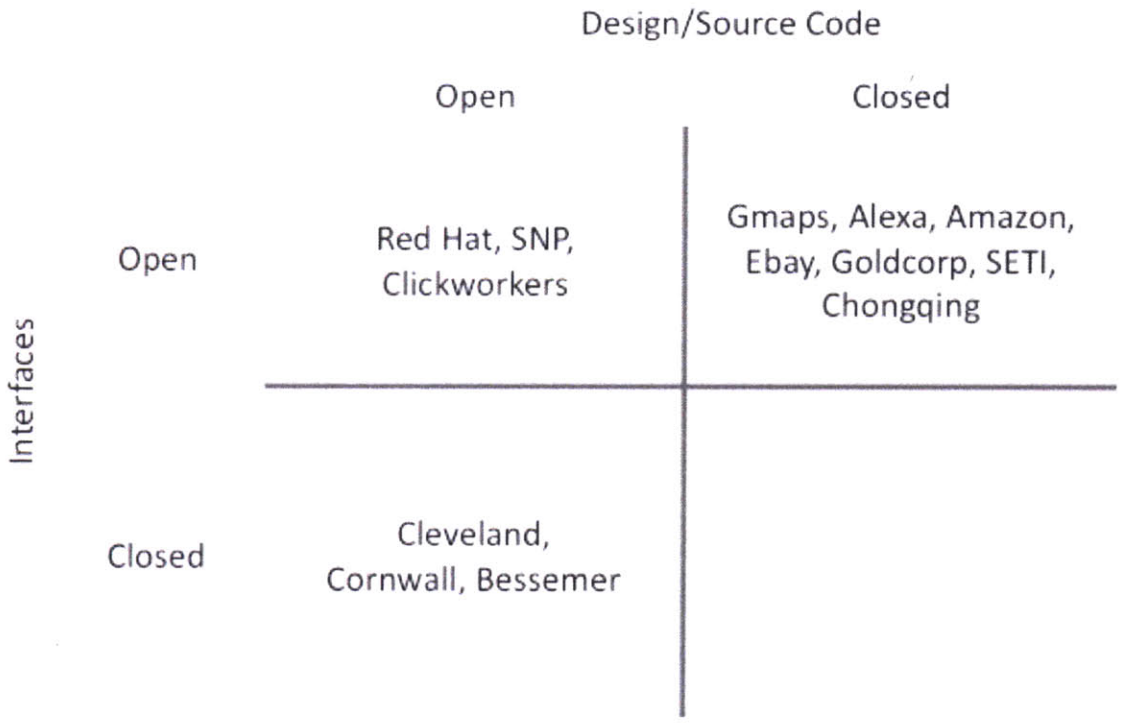


Figure 20: Opening system architectures versus opening designs.

The cases in the bottom left quadrant of Figure 6 involve the free revealing of design information but have closed interfaces because they are highly integral systems for which interfaces are not well defined. In these cases users and developers tweaked parameters within a fixed architecture. The cases in the top left quadrant of Figure 6 involve the disclosure of both designs and interface information. The cases in the top right quadrant employ open interfaces, but keep the underlying designs or data hidden or proprietary. SETI is placed in this category because the data that is processed by the computers is not visible or used by the general population.

The matrix illustrates two important points. First, one must be careful to distinguish whether they are employing an open architecture strategy or an open design strategy. While in retrospect this might be evident (Windows is an example of an open architecture whereas GNU/Linux is an example of an open design) the distinction is not often made explicit within the literature on open innovation. To the extent that academic camps discussed in Chapter 2 ignore one or the other of these quadrants due to definitions (user innovation versus open innovation), they

miss the broader picture. To the extent that creation of an OCSD can be treated as a design problem, there is considerable leeway whether interfaces or design information is revealed, and how much of each is formally defined.

Second, the results of this matrix demonstrate that a wide variety of such strategies could indeed be sustainable depending on the goals of the sponsor/architect. The operative question involves how a sponsor can define standards and identify categories of design information to be released.

3.6.6 Standards in OCSD

Many authors discussed in Chapter 2 suggested that a key challenge to creating an OCSD involves coordinating the efforts of diverse, distributed developers. Standards are an important means of achieving this coordination in the absence of a price signal or the hand of management. Table **8** compares the standards that played the greatest role in facilitating coordination in each mini case, together with what was shared and how modular the technology was. Regimes in which interface standards played a prominent role are highlighted in yellow; others are highlighted in blue.

Table 8: Comparing what was shared, the modularity of the technology, and the important standards.

	Technology		
	What Was Shared	**Level of Modularity**	**Important Standards**
Cleveland	Furnace design & Performance data	Low	Measurement; Architecture Description
Bessemer	Component desgns; best practices	Low	Measurement; Architecture Description
Cornwall	Design & Performance data	Low	Measurement; Architecture Description
SNP Consortium	SNP Database Information	Semi-Modular	Interfaces; Data Formats
Chongquing	Architecture Blueprints	Semi-Integral	Interfaces; Architecture Description
Red Hat GNU/Linux	Source code for GNU and Linux Kernel	Semi-Modular	Interfaces; Data Formats
API Model: Google, Ebay, Amazon	API, Some Functional Modules	High	Interfaces; Data Formats
NASA Clickworks	Databases	High	Interfaces; Data formats
Goldcorp Mine	Mine Data	Semi-Modular	Data formats; Measurement
SETI Project	Radio Data	High	Interfaces; Data formats

The cross case comparison allows us to list the different kinds of standards that play a role in any OCSD analyzed:

1. Measurement Standards;

2. Architectural Description;
3. Data Formats; and
4. Interfaces.

For this analysis, *measurement standards* include units for describing performance or design information for the technology in question. For example, in the Cleveland case consultants and engineers exchanged information on the chimney height, flu-gas heat, and efficiency of the furnaces. For comparison across different furnaces this information needed to be standardized. *An architecture description standard, on the other hand, refers* to the units used to describe the design itself. For example, the Cleveland case utilized a fixed furnace architecture, which needed to be copied across different use cases to create useful comparisons. In many cases where the complete architecture was copied, the measurement standards and architecture description standards were most important for coordinating activity in the OCSD. Data format standards encompass file formats for databases. Finally, interface standards refer to data exchanged between elements of an architecture, and to physical interfaces between elements of an architecture.

Table 4 shows that, as expected, interface standards only play a prominent role for the modular technologies. For the integral technologies the complete architecture was usually shared making interfaces less important, while making descriptive and measurement standards more important across the community of developers. However, these dichotomies are not mutually exclusive. For example, in the Chongqing case the entire motorcycle product architecture was shared, yet interface standards also played a critical role in coordinating the activity of diverse subsystem developers.

3.6.7 Value Chain Analysis

One can begin to identify categories of standards in more detail by examining the architecture of the value chain in which the various technologies reside. For each mini case a schematic of the high level structure of the value chain has been created. Systems can be divided between "downstream" and "upstream" depending on their

relative stage in the value chain (Christensen, Raynor et al. 2001). This distinction works for both assembled consumer technologies, such as motorcycles and computers, or for business-to-business technologies, such as those in mining or biotechnological information. In the former, "downstream" implies lower levels of abstraction, such as components and hardware versus software and applications. In the latter, "downstream" suggests being an input to the production process of another company.

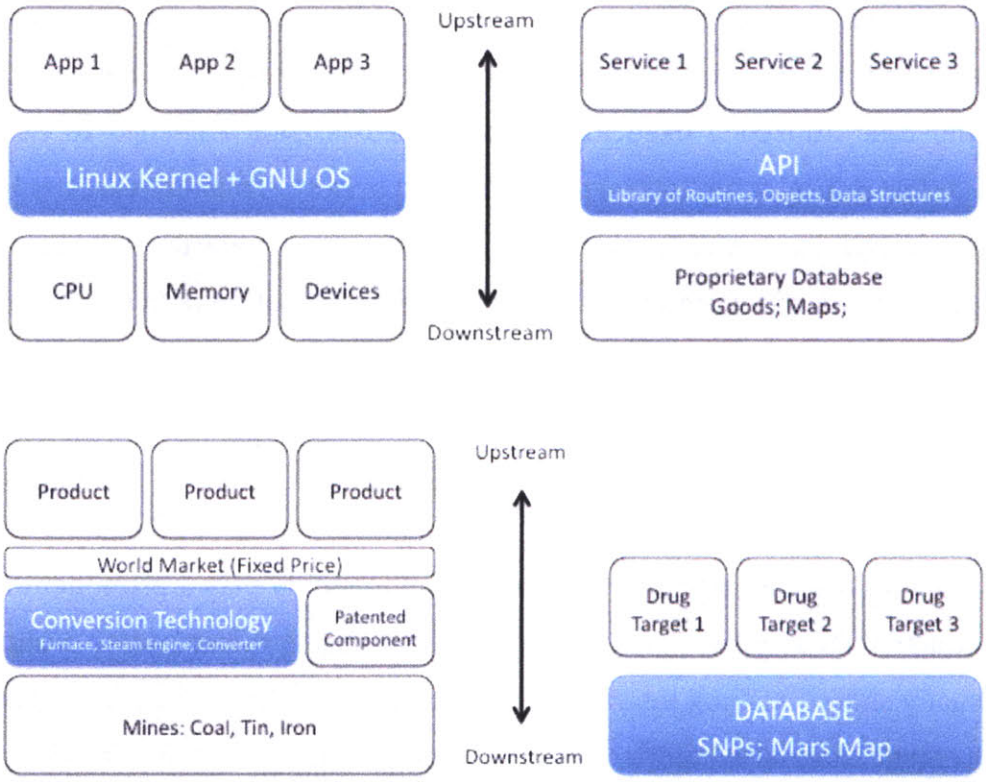


Figure 21: Value chain decomposition of the Linux ecosystem (top left), API ecosystem (top right), Mining industry (bottom left), and database examples (bottom right).

The high level schematics in Figure 7 are, of course, somewhat subjective. But they also highlight the relationship between opening design and common distinctions made by economists between vertical and horizontal layers in a value chain.

Each horizontal layer represents a given stage in the process of material or information transfer. The figures make clear that the "open" element within a value chain can either occupy an entire layer or one part of the broader layer. Further, when discussing interface standards one must be clear to distinguish between those

that facilitate decoupling of layers (whether layers of abstraction or layers of materials processing), and those that facilitate decoupling within layers.

For example, all of the components within the motorcycles made in Chongqing exist at a fixed layer in the value chain. The interface standards are therefore horizontal in that they enable components within the layer to intersect. APIs, on the other hand, enable decoupling of work between layers of the value chain (user facing applications versus lower level databases and routines). The level of detail in these mini cases does not enable analysis of the impact of standards design for horizontal versus vertical standardization beyond speculation. This is analyzed in more detail in the next chapter.

In short, we must add the following distinctions to the kinds of interface standards present:

1. Vertical Interface Standards between layers in the value chain; and
2. Horizontal Interface Standards within layers in the value chain.

If we combine this insight with the distinction between sharing architecture and sharing design information, we can begin to more precisely define the strategic choices faced by companies operating within a value chain. Figure **22** presents a schematic for these basic distinctions. It highlights the difference between horizontal and vertical interface standards in a value chain, as well as the difference between sharing interface standards only and sharing source code or design information at a given layer.

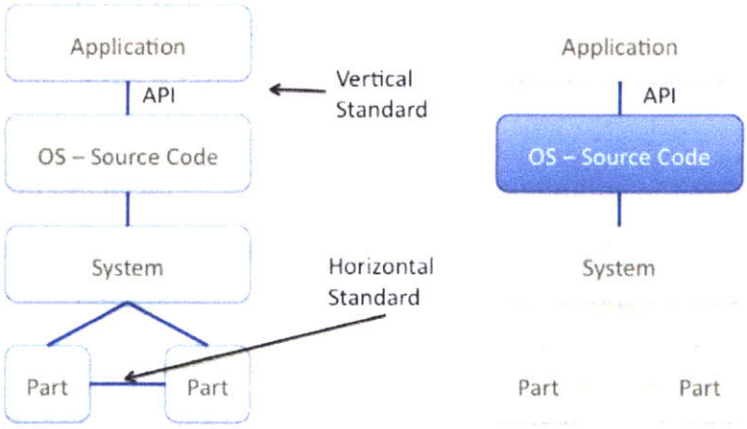


Figure 22: Combining distinctions between architecture and source code, and vertical versus horizontal interface standards. This schematic illustrates the case of the computer industry. Shared information is shown in blue. Proprietary information is shown in yellow.

3.7 Chapter Conclusions

3.7.1 Summary of Correlations

As a whole, the cross case analysis revealed important similarities and differences between the thirteen mini cases. A limited set of three goals explained sponsor goals for all thirteen cases. More specifically, opening accomplished these goals by lowering transaction costs associated with designing new or complementary systems and therefore increasing the pace of innovation. In every case, opening design to third party collaborators was accomplished in the face of either technological uncertainty or market/user need uncertainty. However, none of the cases involved both low or both high technological and market uncertainty.

The kind of uncertainty present in each case further correlates to the kind of innovation encouraged within the open regime. Where technological uncertainty was high (and market uncertainty low) incremental innovations where developed and shared across the community. Where market uncertainty was high (and technological uncertainty low) architectural innovations were carried out by third parties and connected to existing systems. In retrospect this finding may be, at a certain level, a matter of definitions. Quite often architectural innovations are developed to address new market needs. Therefore, one would expect incremental changes to address fixed needs, or architectural innovations to address variables

needs. However, this is not a hard rule. Architectural innovations could certainly be used to address fixed needs. At the very least, the correlation suggests an important tie between the sponsor goals, constraints, and the kind of innovation encouraged. It thus merits further investigation.

The cases also reveal important differences in the architecture of the technologies being shared. Contrary to assertions in much of the literature, modularity of technology alone is not sufficient to explain incentives in an open regime. The cases show great diversity based on whether interface standards/information are shared and whether designs are shared. This suggests that we must distinguish between open architectures and open source within OCSD. Finally, a value chain analysis of each case suggests that we must examine the relationship between layering and opening. In particular, each case involves opening a specific layer of the value chain, whether the sponsor played a role within the open layer or at an adjacent layer. This suggests that the difference between vertical interface standards (between layers) and horizontal interface standards (among components in a layer) should be further investigated.

The findings validate and enable further development of the framework defined in previous chapters. Importantly, the correlations suggest that there are three primary strategies pursued by sponsors within open regimes. These strategies are, defined by unique combinations of goals, constraints, kind of innovation, and kind of standardization. These three strategies are expanded upon in more detail in subsequent chapters.

Despite the correlations identified, some important caveats should be made clear. The findings should not be interpreted to mean that OCDS will always increase the pace of innovation, or that patents and proprietary regimes necessarily slow innovation. In fact, each case included some form of patenting or proprietary development. Thus, at a high level the cases demonstrate a spectrum of activity, in which opening interfaces and designs in some areas serves as a kind of lubricant to

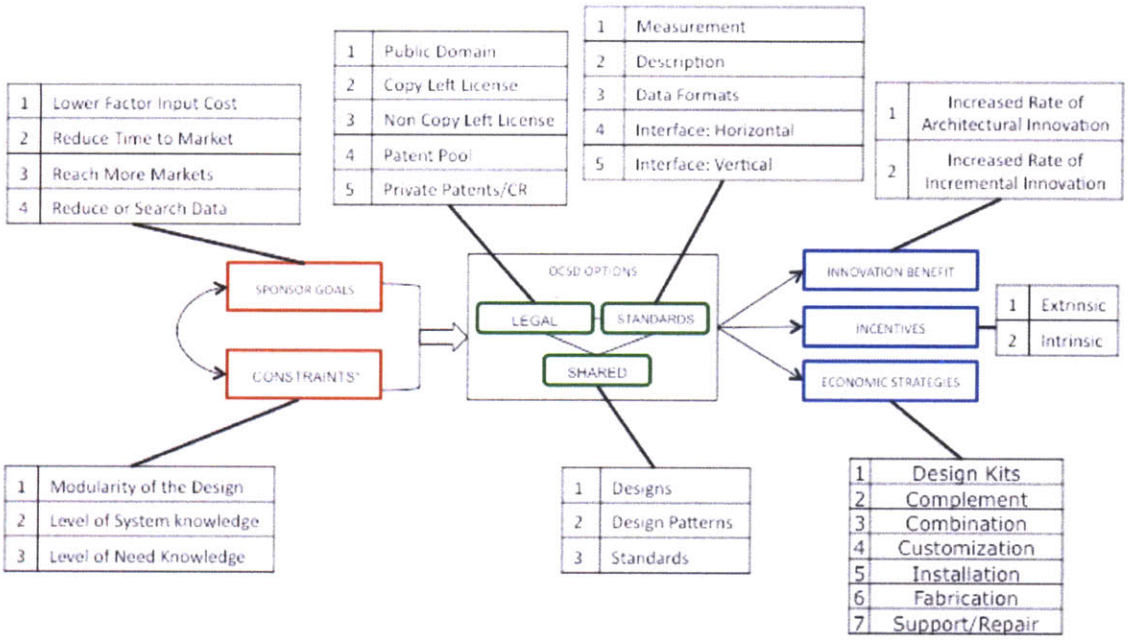
the innovation process. Another way to state this is to say that open standards and open designs create a kind of *neutral space* which can attract third party proprietary and non-proprietary efforts, thus making a technological ecosystem compete more effectively with another ecosystem.

Finally, while important similarities and differences were identified between and among cases, and distinctions made, it is important to note that the cases themselves are all rather different. One of the most critical differences, not examined in depth here, was between hardware and software systems. This difference was glossed over, in part, in an effort to suggest that at the appropriate level of abstraction one could identify common principals behind open, collaborative system design. Moreover, many of the cases involved a mix of hardware and software.

However, substantial contextual, cultural, and technologies differences underly the cases. Decisions made to share furnace design data in the mid 1800s, and the API-based strategies currently employed by companies like Google are made in very different environments. This chapter purposely overlooked such differences in order to extract common factors underlying the OCSD process. While this is necessary and valuable further analysis of the difference between contexts can always help illuminate the scope and limitations of conclusions presented here.

3.7.2 Combining Variables and Constraints: Building the Framework

A number of distinctions pertaining to modularity, architecture, and uncertainty have been made via analysis of the mini cases. We now examine how these contribute to and fit within the broader framework articulated in Chapters 1 and 2. Figure **23** takes the different distinctions and options identified in this chapter and adds them to the framework created in Chapter 2.



* Constraint associated with the system whose open development is being planned

Figure 23: Framework with taxonomic categories fully formulated.

3.7.3 *System Drivers: Performance & Function, Energy & Information*

The correlations identified in this chapter reveal deeper rifts with respect to the options added to the framework. In OCSD where the systems being developed were performance constrained, open regimes followed a very different pattern than OCSDs in which the systems were market-limited (that is, where the number of end uses was unclear). With some important exceptions, this dichotomy fits with a number of broader distinctions described in this chapter (see Table 9 below), such as the sponsor type, primary innovation goal, what is opened, and whether the system primarily processes energy or information.

Table 9: Correlating sponsor-type, knowledge, innovation goal, system-type, and opening. Three exceptions to the correlations are highlighted in red and discussed in more detail below.

	Case Name	Sponsor Type	System Knowledge	Market Knowledge	Primary Development Goal	What does the System Transform?	What is Open?
1	Cleveland	User	Low	High	Better Performance	Energy	Full Design
2	Cornwall	User	Low	High	Better Performance	Energy	Full Design
3	Bessemer	User	Low	High	Better Performance	Energy	Full Design
4	NASA Clickworkers	User	Low	High	Better Performance	NA (Database)	Data
5	SETI Database	User	Low	High	Better Performance	NA (Database)	Data
6	Goldcorp Mining Co	Developer	Low	High	Better Performance	NA (Database)	Data
7	Alexa Search	Developer	Low	High	Better Performance	NA (Database)	Data
8	SNP	Developer	Low	High	Better Performance	NA (Database)	Data
9	Chongqing	Developer	High	Low	**New End-Uses**	**Energy**	Interfaces & Modules
10	Red Hat	Developer	High	Low	New End-Uses	Information	**Full Design**
11	Google Maps	Developer	High	Low	New End-Uses	Information	Interfaces & Modules
12	Ebay Developers	Developer	High	Low	New End-Uses	Information	Interfaces & Modules
13	Amazon Developers	Developer	High	Low	New End-Uses	Information	Interfaces & Modules

The table illustrates that where system knowledge was low (orange and green), sponsors sought better performance of a fixed architecture, by revealing the complete design or all of the data; where market knowledge was low, sponsors sought architectural innovations for new functions or end uses, by sharing interface information and some modules (blue). The orange and green highlighted sections distinguish cases in which design information was shared versus cases in which data was shared.

The distinction between cases in which sponsors sought designs with improved performance and those in which sponsors sought new functions correlates partially – though not fully – to whether the system processed information or energy. For example, in the Cleveland case the blast furnaces transformed energy because the input to the furnaces was coal, which was burned for its heat, which was used to transform iron-ore into pig iron. The web services examples like Google Maps, by contrast, transform user inputs into useful information via the use of databases.

However, based on the data analyzed in this chapter, the distinction must be qualified in two important ways. First, the cases in which only data bases where shared should be considered information intensive, yet are closer in their correlations to the energetic cases than the software cases. For example, in these cases the sponsors had poor knowledge of the system (the data and its correlations), but high knowledge of its end use. The sponsors sought what is best described as incremental improvements to the database itself, rather than new functions. The database cases should therefore be considered a third basic approach to developing OCSD, as discussed in more detail in the next section.

A second exception to the information-energy distinctions involves the case of Chongqing. Chongqing is a case in which the sponsors sought new markets through new functionality and performance. Yet, as highlighted in Table 9, the system primarily transformed energy (motorcycles convert chemical energy into kinetic energy). Why? As the description of this case clarified, the motorcycles are low performing. That is, the OCSD sponsors consciously accepted lower performance in order to use a more collaborative development approach that increases the speed with which new designs are brought to new markets. This exception is therefore important to our overall understanding of OCSD. It suggests that performance versus function is a deeper dichotomy than energy versus information.

The third exception highlighted in Table 9 involves the fact that for Red Hat GNU/Linux, the entire design is freely revealed. In all of the other cases in which new markets are sought, only the interfaces and *some* of the design are revealed. This exception was discussed in detail in section 3.6.1 above. As describe in that section, the GNU/Linux case is actually mixed. GNU and the Linux Kernel was originally started by users, not by for-profit developers like Red Hat, with a specific goal of creating a working operating system. The OCSD regime has since morphed into one in which for-profit developers like Red Hat sponsor development. In

moving to the new regime, a number of proprietary application module have indeed been added to the GNU/Linux stack.

3.7.4 *Using the Framework to Guide Strategy*

More broadly, the cross-case analysis suggests that some combinations between options in the framework are mutually exclusive. In no case did sponsors have poor knowledge of *both* the system design and the end use. Further, sponsor knowledge correlates to three main sponsor sub-goals that encompass the entire thirteen cases: (i) increase market diversity (ii) decrease factor input costs, and (iii) efficiently search a database of information.

Table 10: Three Strategic Bundles for Opening

		Suggested Inputs			Suggested Decisions		Likely Result
	Name	**Typical Sponsors**	**System Knowledge**	**Market/Need Knowledge**	**Principal Elements Shared**	**Important Standards**	**Type of Innovation**
1	**Incremental Development**	Users	Low	High	Entire Designs	Measurement, Description	Incremental
1.A	**Database Development**	Users or Developers	Low	High	Data	Measurement, Data Formats	Incremental Pre-Competitive Research
2	**Architectural Development**	Developers	High	Low	Design Patterns, Interfaces	Vertical Standards, Data Formats	Architectural

Table **10** groups these "strategic bundles" into three categories. The first involves users of the technology seeking to improve performance of a fixed architecture with a specific end goal by sharing complete design information. The second is a variant of this first case – in it users or developers share data in order to perform incremental, pre-competitive research with a specific end goal. The second strategy involves developers that have a good understanding of the technology, but poor understanding of the all of the potential end-applications. In these cases they sought architectural innovations to identify new functions and new markets, by sharing interfaces.

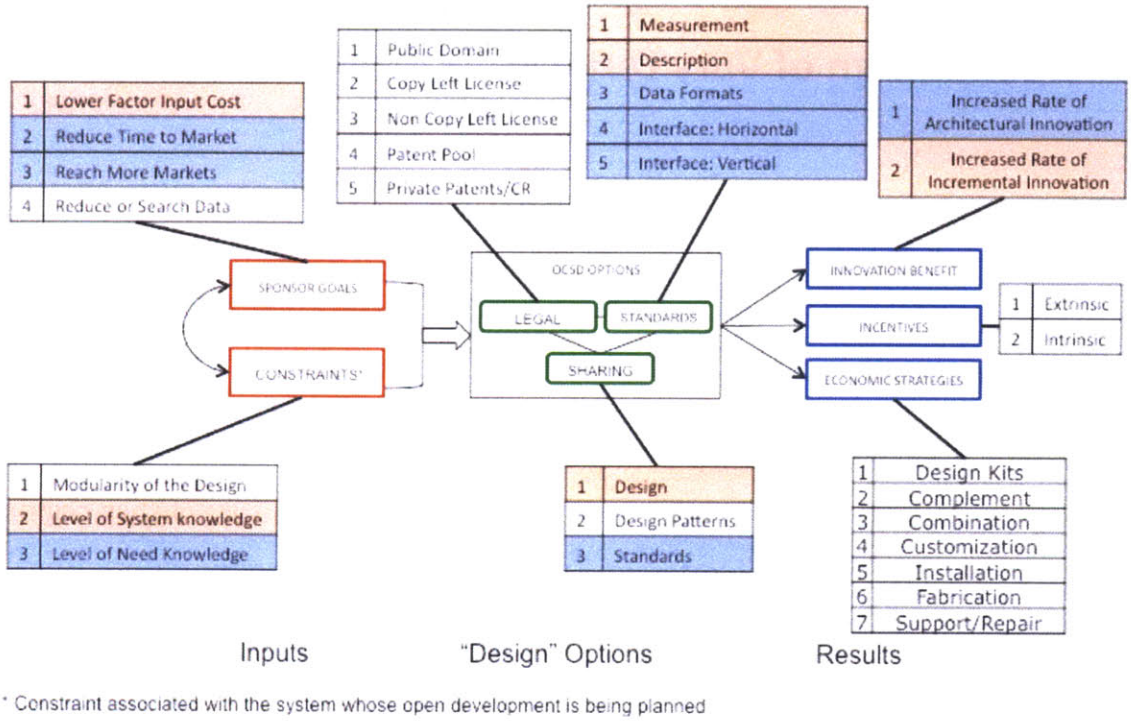


Figure 24: Conceptual Framework Including the Strategic Bundles Associated with Architectural and Incremental Development (Blue and Orange, respectively).

Figure **24** presents the conceptual framework with strategic bundles #1 and #3 highlighted in orange and blue respectively. Shared data is ignored in the figure. The correlation between these elements in the mini cases suggests that successful, sustainable OCSD should include only the combination of elements of similar color. In many cases there are, of course, significant decisions that must be made within each taxonomic category. Further, the correlations are lose rather than hard. For example, some kinds of standards must be shared in the strategies even if strategy #2 is taken. However, the coloring suggests areas to focus on in developing such strategies. These will likely drive the development of the regime.

The question remains whether these basic strategies, and the new taxonomic concepts, are validated by an OCSD not used to create the framework. Also, how might such a framework be applied within the context of biotech and the non-IT industries? These questions are addressed in the succeeding chapters.

4 Open Design of Very Large Scale Integrated Circuits

4.1 Introduction

Chapters 1 through 4 examined multiple dimensions of open collaborative system development (OCSD) both to limit the space of choices for sponsors of such regimes and to create a framework for design. Two important classes of questions remain. The first class involves validation: to what extent do the distinctions that were *not* examined to create the framework encompass OCSD; or to what extent do the strategic bundles described at the end of Chapter 3 exclude a closed/proprietary approach? The second class of questions involves application: how can potential sponsors use the framework to make decisions given a set of goals and constraints?

This chapter addresses the first set of questions through an in depth examination of one case: the opening of the design of very large scale integrated circuits (VLSI) by Carver Mead and Lynn Conway in the late 1970s. This case was chosen for a number of reasons. First, the opening of VLSI design had important economic implications that continue to reverberate throughout the semiconductor industry to this day. Second, it is an often cited example of how modularity in technology can be used to create new sources of value (Baldwin and Clark 2000). Third, a number of researchers have drawn parallels between the current state of synthetic biology and the semiconductor industry in the 1970s (Carlson 2007). Finally, some aspects of the open regime failed, most notably the efforts to create a shared library-based design. As a whole the case provides a rich context in which to deepen the framework and to explore both strategies. Further, it clarifies how some aspects of the second strategy can fail, without necessarily dooming the process.

The chapter is organized into five main sections. The first introduces the case. The second discusses technical, organizational and cultural motivations for attempting to open the technology. The third section analyzes Mead and Conway's innovations, with particular focus on the role of standards, openness, and the notion of "separating design from fabrication." The fourth section analyzes the way in which Mead and Conway's advances were validated and diffused through a student "multi-chip" project. The fifth section draws conclusions from the case. Data for these sections is derived from personal interviews, primary source material such as memoirs, contemporary academic articles, and secondary historical material.

The opening of VLSI validates the proposed framework. It is an example of the second strategic combination identified in Chapter 3, in which a well understood technology has myriad unknown market applications. As such, vertical and horizontal interface standards were used to encourage third party design and tap significant market opportunities that would have otherwise been very costly for one company to pursue. The case also reveals the role of different kinds of standards more clearly. Both vertical and horizontal standards, as defined in Chapter 3, are important. Yet it is clear that in this case vertical standards, enabled by new measurements and descriptive standards play the decisive role in opening the technology. Finally, the case highlights the socio-cultural and organizational difference between an open and closed value network. Because it optimizes against fundamentally different criteria, favoring speed of innovation over precision and performance, the move from closed to open value faces significant socio-cultural challenges and assumptions embedded within the structure of organizations. This suggests that starting in a closed regime and moving to an open regime is very difficult without the creation of a wholly new organizational structure and culture.

4.2 Background: Very Large Scale Integration

In early 1976 Carver Mead, a professor of Computer Science at the California Institute of Technology, gave a presentation at the Xerox Palo Alto Research Center (PARC), the preeminent locus for innovation in computing technology worldwide, at the time. In his speech, Mead stressed the need for new methods to design integrated circuits. On the horizon were silicon fabrication techniques that would enable tens of thousands of transistors to be placed in proximity on a chip. In 1976 the limit for chip design involved hundreds of transistors and was known as Large Scale Integration (LSI). These new techniques would bring semiconductor design to a new level of very large scale integration (VLSI). In VLSI each subsystem alone represented an LSI problem.

Mead's presentation was met with muted interest by the PARC researchers (Hiltzik 1999). Many were experts in the established practice of LSI and assumed the methods could be extended. Others were skeptical of Mead's fundamental hypothesis that was that thousands of transistors could be fabricated to operate in such close physical proximity in the near term. The talk did, however, spark the imagination of Lynn Conway, a PARC veteran and accomplished designer of mainframes who had recently been frustrated by the slow pace of prototyping LSI chips. Mead and Conway spoke following the presentation and soon after decided to launch a collaboration that would change the structure of the semiconductor industry.

Over the ensuing years Mead and Conway's VLSI design methods would spark a sea-change in the semiconductor industry.[13] Their "structured design methodology" was perhaps most famous for its incorporation of simple "design rules" with which computers could easily articulate otherwise highly complex sets of constraints, making computer aided design possible. The pair also created a foundational VLSI

[13] This is not to say that their design methods fully caused the change. Rather, they were an important advance that, as described later, certainly helped catalyze and foster rapid innovation.

design textbook, helped developed hundreds of courses, and orchestrated a distributed "multi-chip" design project that definitively proved that distributed teams could conceive and construct VLSI systems at a fraction of the time and cost of established methods. Their methods led directly to the creation of numerous multi-billion dollar companies, many of which seized on its simplicity to operate fully "fabless," meaning that the companies designed microprocessors to be fabricated by others (NRC 1999).

The net result of Mead and Conway's activities was to open the design of complex integrated circuits by increasing access to the core concepts, revealing otherwise proprietary standards, and facilitating a clean separation between knowledge needed by designers and fabrication engineers. The latter, as described below, was enabled by publishing data format standards, and by developing a new measure with which to separate the description of constraints in the design from the native constraints in a given fabrication facility. In the language of computer engineering, Conway and Mead made the designs "portable." And as with all questions of portability, it was achieved at what initially appeared to be the expense of performance.

Mead and Conway's efforts resemble current goals of synthetic biologists in a number of ways. At a pragmatic level, while the exact physical challenges differ, both VLSI and synthetic biology are concerned with designing and testing systems of highly complex and coupled networks. This means finding ways to abstract and simplify design, to formulate standards, and to develop computational tools that can articulate highly complex constraint sets. More generally, these difficulties lead both fields to see the value in separating the challenges of design from the challenges of fabrication. For VLSI this meant creating a simple interface between designs and fabrication facilities (fabs). For synthetic biology the challenge is more complex. If fabrication is defined as the creation and replication of DNA, it means finding techniques that can ensure translation and transcription regardless of the end goal. Standard cell lines, assembly standards, and related techniques must all be

developed. Nonetheless, the basic similarities in goals, such as standardization, sharing circuit libraries, and computational design tools, are noteworthy.

4.3 Moving From Factory Design to Urban Planning

To appreciate Mead and Conway's advances the rudiments of integrated circuit design must first be understood. An integrated circuit is a network of transistors patterned on a chip. The topology of these networks determines the logical operations performed when electrons are shuttled through the network. The ultimate goal for circuit designers is to create patterns that perform various operations at maximum speed. Loosely speaking, designers trade topologies with better logic speed and functionality against delays created by circuit layout, within constraints caused by manufacturing limitations (Mead and Conway 1980). The latter are caused both by physics based phenomena, such as current leakage and impedance matching, as well as basic chip fabrication limitations.

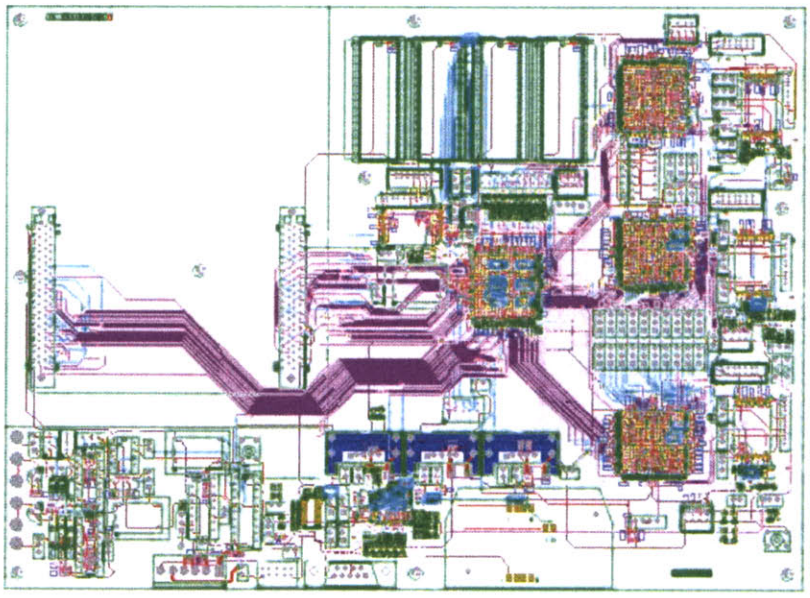


Figure 25: VLSI Circuit Layout. (Courtesy of: www.roshinimicro.com)

The design process thus involves specifying a logic design and then finding a layout for this logic design that maximizes speed while fulfilling the constraints. Often a given layout will be inadequate forcing a return to the logic design level. Depending

on the complexity of the circuit and the structure of the design process, these iterations could become very time consuming (Mead and Conway 1980).

When he presented to Xerox PARC in 1976 Mead was concerned with how an engineer would eventually manage the hundreds of thousands of tradeoffs associated with patterning thousands of parts. The burden was heightened by the fact that as transistors shrank, the layout of the wires became as important for performance as the order of the transistors (Hiltzik 1999). While in the LSI regime engineers were primarily concerned with the order of transistors. In VLSI they would now also have to worry about the length of the wires. This effectively multiplied the "trade space" to be explored, adding tremendously to the complexity of the problem. As Conway would later remark, if previous technologies forced designers to think like factory planners, taking in resources at one end and converting them in a series, VLSI "forced you to think like an Urban Planner....you had to think hard about where the roads go." (Hiltzik 1999) Mead referred to this challenge as "the problem of complexity."

For Mead the management of complexity was also hindered by problems associated with expertise. As chips became more complex individuals were assigned sub-functions such as logic design, and later, even sub-sub-functions. These decompositions became a useful method to manage complex information. However, they greatly hindered flexibility in the overall design because no one person could grasp the overall impact of all of the functions. As he put it, "the specialization of work was a big hindrance to people doing system chips. [Big companies like IBM and Xerox] had broken the design process down in a German army fashion and everyone was precise and nobody had their head around the whole thing." (Mead 2009)

Having consulted for industry, Mead also became aware of two important facts that would help break down this "German army" of expertise – the difference in skills sets between fabrication and design, and the similarity between the design rules (or

constraints) of different fabs across the industry. He notes: "we worked with people that had fabs. That led to the notion that the set of skills to run a fab were so different than those to design complex systems. It also turned out that everyone had something called design rules to ensure the transistors would operate. So compared to understanding how to run a fab [these design rules] were simple." These observations led Mead to conclude that new methods might enable designers to see the whole problem by ignoring fabrication and by simplifying the design model. These new designers might be able to work at multiple levels of abstraction to make system level trades. "We came up with the concept of the Tall Thin person – someone who could span all levels." (Mead 2009) By levels, of course, Meade meant layers of abstraction from circuit layout up to sub-logic design and complete logic design.

As a practicing engineer, Conway was particularly sensitive to the practical problems of trying to get the German army to create new prototypes quickly. Prior to collaborating with Mead, Conway had been asked by Xerox to build a special purpose image processing computer (Hiltzik 1999). Her efforts were successful, but the resulting design was clumsy and not economical due, in part, to a need to kludge together the standard LSI processors not optimized for her application. As she noted, "I was working on special purpose architecture for image processing at the time. I had become aware that there was a gap between the sorts of systems we could visualize and what we could actually get into hardware in a timely way." (Marshall, Waller et al. 1981)

For Conway, the gap between what could be conceived and what could be created quickly and economically was due to the problem of excessive division of design labor. It also had its source in the culture of secrecy in industry and the poor communication that resulted. As Conway writes:

"At the time, the industrial world of system designers and IC designers was fragmented into a vast array of independent, competing clans having very different practices. Design groups specialized in different market application areas, and were further divided by the technology of implementation (nMOS, CMOS,

etc). Industrial secrecy had fostered local craft practices, and cultural drift had produced wide gaps between firms. Within each clan, expertise was split by divisions of labor such as system architecture, logic design, circuit design, and layout design. As a result, most architects were unable to understand layouts, and most layout designers unable to understand the system-level functions of chips, even within their own domains of application and technology." (Conway 1982)

Conway lamented the craft-like and highly specialized structure of knowledge in the design environment because it affected her ability to develop computer systems rapidly around new integrated circuits. Thus, even in the LSI regime of hundreds of transistors it was very hard to co-develop a new computer system with a new chip in a short time period. The expertise needed was overly fragmented and the culture secretive.

The problems associated with developing new designs also extended to manufacturing the chips themselves. Microchip fabrication consists of creating a "mask" which is used to etch circuit layouts onto wafers (Mead and Conway 1980). Tremendous resources had been dedicated over the years to perfect these processes. The Mask process had the great advantage of being "pattern independent", meaning that a given fab could make any chip, once the proper mask was created. However, because each fabrication facility had different constraints on design parameters such as "line resolutions," the creation of these masks was particular to the fabrication process. Professional circuit design thus required iteration and fine tuning between the specific fabrication process being used and the design itself, which required a significant amount of communication between the two groups. Design, in short, was not separated from fabrication.

The practical effect of this structure of knowledge and the broader industrial culture was that even at the LSI level of complexity transaction costs associated with exploring new designs were very high. Firms could bring new standard chips to market at reasonable costs. But *prototyping* circuits was time consuming, expensive, and required collaboration among many outside experts. Conway recalls, "At that time, and even now in most integrated circuit design environments, the mask making and wafer fabrication required to implement prototypes for a design project

cost about $15,000 to $20,000, and with some luck took only three or four months getting through the various queues." (Conway 1982)

In short, while Mead was concerned with the theoretical problem of developing design methods for highly complex systems, Conway was motivated as much by practical problems experienced when she prototyped systems. She attributed these problems to the fragmented and specialized knowledge base, a secretive industrial culture, and the interconnection of design with fabrication due to fab specific design constraints. Both Mead and Conway's concerns seem to be symptomatic of the common underlying problem of managing highly complex design information. Both resulted in the same goal – simpler design methods for VLSI. Thus " a shared vision began to develop in [the] team – not just Carver's original 'theory about...the challenge of complexity' – but my vision and approach on how to...craft simplifications that had a good impedance match with the knowledge base of the average digital designers of the time." (Conway 1999)

4.4 The New VLSI Methods

After a period of intense discussion and collaboration, the pair hit upon a set of methods that tremendously simplified the VSLI design process. What specifically about it was innovative? Without delving into excessive technical detail, a few elements are important to the discussion. These included removal of the need to iterate substantially between logic design and circuit layout through the creation of methods, which enabled design to be carried out at a high layer of abstraction. The methods made use of a dimensionless unit – lambda – which Conway invented to describe design rules in a fab independent way, as well as the creation and diffusion of interface standards, and the development of computational tools. We might say, then, reduced the barriers to innovation and simplified design by *reducing the amount of knowledge* required by designers and by *restructuring the form of that knowledge (Conway 1982)*. This was achieved through the management of layers of abstraction.

The method reduced the amount of knowledge required by designers in two principal ways. First, it omitted myriad design choices that were not completely essential to the functioning of the chip. This meant, for example, insisting on only one kind of standard element for a certain function, where expert LSI designers would have a choice of hundreds. Conway writes, "we would use only simple two-phase clocking...use simple dynamic registers...teach basics at the circuit level to enable transistor ratio calculations...use PLA's for any messy logic and for control logic...use 'stick diagrams' for initial system cells to layout conceptual designs." (Conway 1999) In other words, Mead and Conway sought to distill only the key functions necessary for creating a circuit, emphasizing the use of only one or perhaps two devices for each function. This standardization of sub elements removed control, but it also hid information that might confuse new students.

The second way in which the designs reduced the amount of knowledge pertains to the development of standard design rules. It is subtler and probably more important. As noted above, an important trade off when designing VLSI chips involves how to increase compactness without violating limitations caused by current leakage, impedance mismatches, and other fundamental physical problems. These constraints could be articulated as simple heuristics for minimum allowable widths, separations, extensions, and overlaps of geometrical objects on the chip (Mead and Conway 1980). And the challenge resulted in the need for iterative design between the logic elements and the circuit layout on the chip. Here Mead and Conway greatly simplified the problem. Rather than creating complex heuristics for each constraint, they hit upon the idea of normalizing all of the rules to one, scale independent, dimensionless unit called "lambda." Because the relationships between constraints were often similar, they could first be defined in units of lambda. Lambda could then be varied to match the resolution of any given fab. As

fabs changed and became more precise, lamda could be scaled down accordingly without changing the basic designs or the knowledge of the design rules.[14]

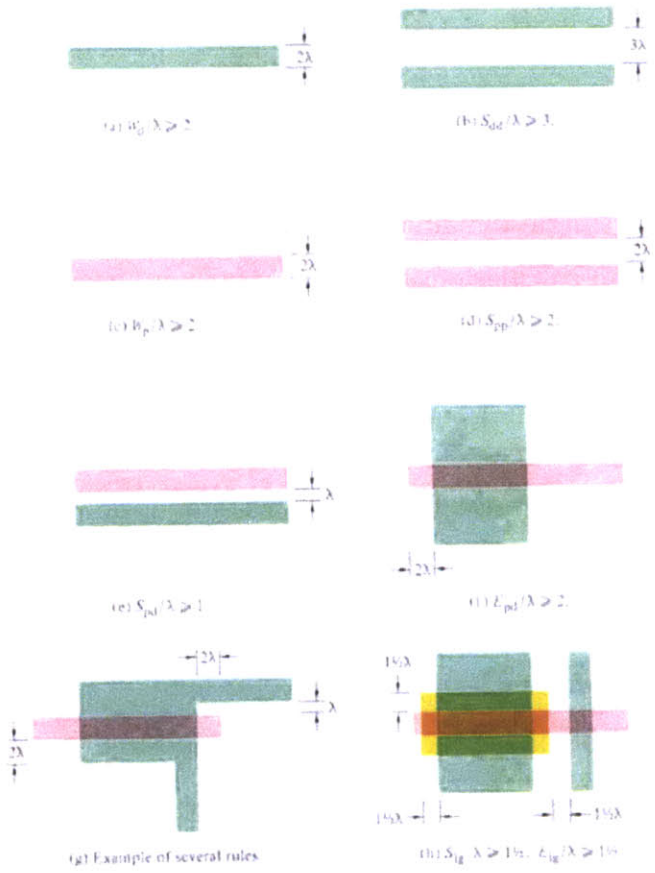


Figure 26: Scale Independent Design Rules Normalized to the Basic Unit *Lambda*. (Source: Introduction to VLSI Systems).

The lambda rules had a major impact on both design and fabrication. Most importantly, they abstracted the description of any given design and its constraints from the native design rules in a fab. In this way, the lambda rules made both the designs and knowledge needed to produce design portable from fab to fab, and avoided problems caused by coordination between rapidly changing design knowledge and rapidly changing fab knowledge. Because they simplified the description of designs, the lambda rules also reduced the computational complexity

[14] It should be noted that a factor of two reduction in lamda yields four times the number of transistors on a chip. Thus, Moore's Law predicts that Lambda will be cut in half every four years, yielding a doubling every two years.

associated with representing and manipulating designs in a computer. This opened the door for the use of simple, computer aided design tools to *test* designs. Automated testing of VLSI circuits was a critical step forward in reducing the time and cost of prototyping.

The lambda rules also enabled the creation of simplified models to guide "the decomposition of problems into sub-problems" and thus enabled design to be carried out at a higher level of abstraction (Conway 1982). As an indirect result, preliminary designs needed much less iteration between logic design and circuit layout, thus removing the need for two kinds of experts in the design process. As Conway wrote, "an individual designer can now rapidly carry a design from architecture to layout in silicon, where previously a team of specialists would have been required." (Conway 1982)

Figure **27** demonstrates the previous LSI layers of abstractions versus the new one implemented by Mead and Conway. Note the conjugation of Logic Design and Circuit Design.

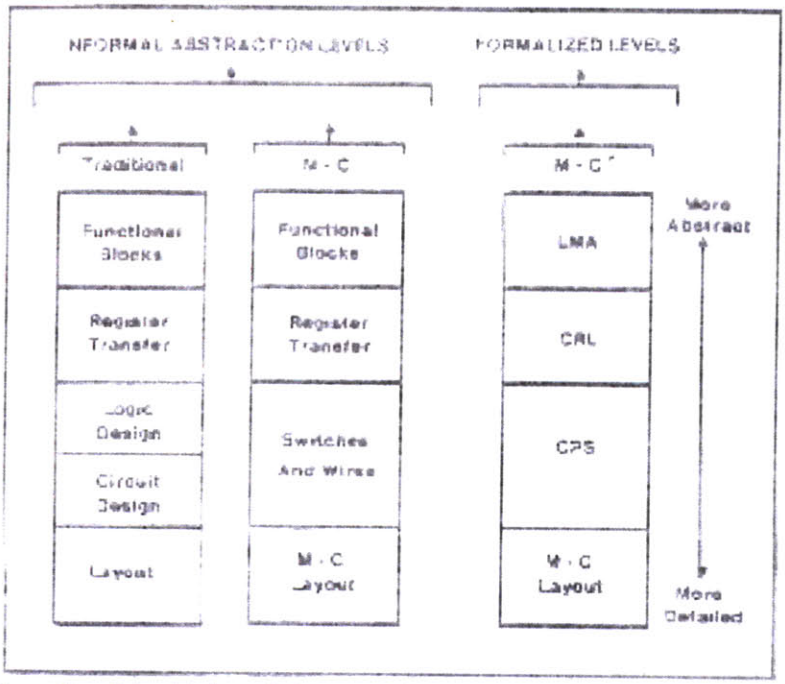


Figure 27: Comparison of traditional informal abstraction levels with the first Mead and Conway approach (M-C) and the formalized approach (M-C') From Conway, 1982

Perhaps most importantly, because lambda was a unit-less measure, the basic approach could also be tweaked to match a wide variety of fabrication processes, thus effectively decoupling the design process from fabrication. This greatly simplified the task of prototyping because it removed the need for communication between fab and design engineers. Importantly, given the tremendous advances occurring in manufacturing technology, decoupling the representation of the design from the details of the fab process ultimately enabled a pool of design expertise to grow independently from the manufacturers. It was one of the critical elements in removing design from production.

Figure **28** below, provides a schematic for how the lambda rules decoupled design from fabrication. It highlights how distributed teams could use design rules to create a chip, test it on a computer, and then send the design to any fab once they were normalized to the proper resolutions. With these advances, designing VLSI circuits was both easier and independent of any given fabrication process.

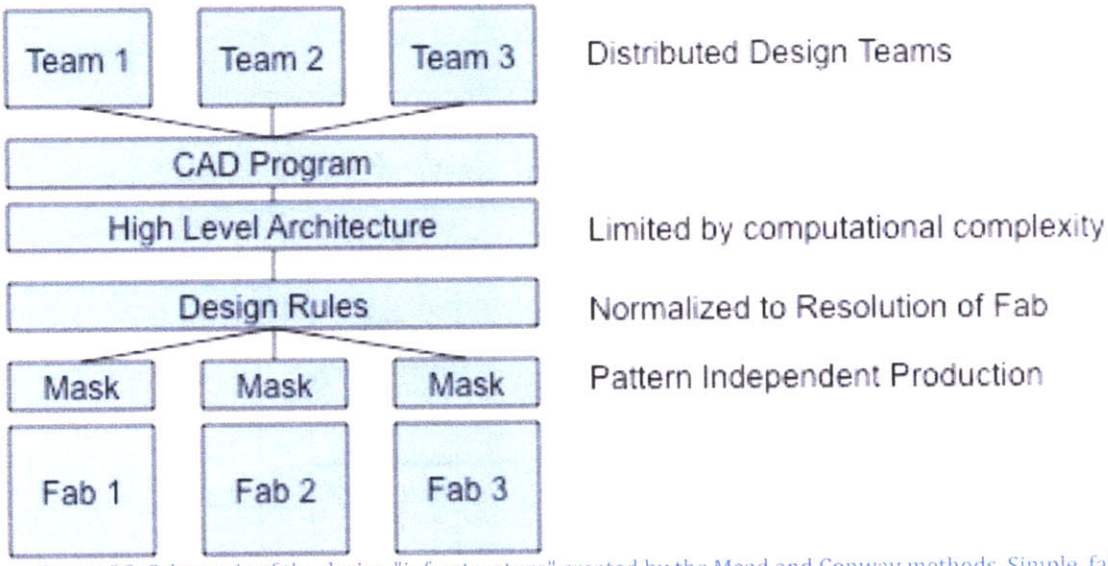


Figure 28: Schematic of the design "infrastructure" created by the Mead and Conway methods. Simple, fab independent design rules de-coupled the representation of the design information from the particular constraints of a given fab. Simultaneously, the simplified representations enabled computers to manipulate and simulate candidate designs more easily. The common and open design language enabled distributed teams to work on "modular" parts of a high level architecture independently. Some of the important elements of this infrastructure, such as the pattern independent production, were already in place.

If these important advances made prototyping easier and cheaper, they were not sufficient for open, collaborative system design. True openness, from an architectural perspective, would require that the rules for multiple processes and interface formats would be publically available. And true openness from a design perspective would further require that at least some elements of the circuits, whether subsystems or libraries of sub-circuits, were shared by a community. While both were the ultimate goal for Mead and Conway, only the former took hold in industry. Once it did, however, it had a major impact.

4.5 Diffusion and Demonstration: The Multi-Chip Project

With the methods mostly developed, Mead and Conway set about on a second phase of their program: to democratize and open design of VLSI circuits. Bucking the norm of secrecy in the industrial community, the team put tremendous effort into diffusing and validating the methods through a textbook, courses, and high profile demonstrations. Most famous among these was the first course taught at MIT in the spring of 1978, which culminated in what became known as the "Multi-Chip" project (MPC79). The Multi-Chip project consisted of validating design methods by rapidly prototyping student designs in a fraction of the time normally needed by experts within the industry.

In MPC79 students were asked to co-design a complex VLSI chip using the simplified methods. Computer based designs could then be sent to a fabrication facility. Software to transmit and compile such designs was created by Allan Bell in Conway's team at Xerox (Conway 1982). The lambda rules ensured that the design files would remain small and performance could be estimated crudely but quickly by a computer.

However, one problem persisted: the design file needed to be sent in a form that could be decoded and understood by fabrication engineers. This would require an interchange format to describe designs. Mead first asked whether a format called

GDSII and owned by the company Calma could be used. However, it was proprietary and the company did not want to release it (Mead 2009). For this reason, Mead and Conway developed a new format termed Caltech Intermediate Form (CIF) to exchange design files. This became a standard in the open design of VLSI chips.

Figure **29** below illustrates the basic design process within the MPC79 project, as illustrated in the original VLSI textbook. It emphasizes that a number of designers in the user community would create and check designs then send them to the fabrication engineers via the Arpanet. The data would be sent via electronic mail as a CIF file.

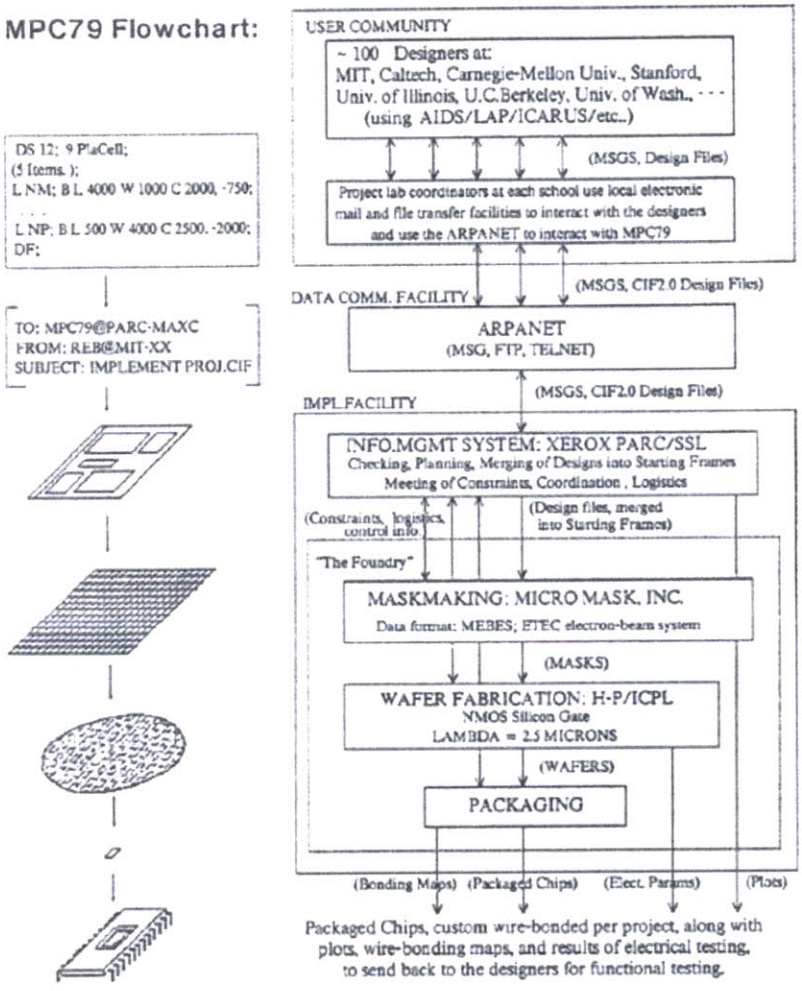


Figure 29: VLSI Design Flow Chart. (Source: http://ai.eecs.umich.edu/people/conway/Retrospective4.html#Success)

The demonstration was risky, especially since Conway had agreed to turn around chip fabrication within a month of design completion, a fraction of the time normally required. Yet it was a tremendous success. Within a few months students formerly unversed in circuit design created and fabricated functioning VLSI circuits in a fraction of the time that experts had been accustomed to. As Conway later wrote:

"By using the implementation system to provide shared access for a large community of users to what amounts to a 'fast turnaround silicon foundry' for rapid mask making and wafer fabrication, we achieved a cost per project on the order of a few hundred dollars, and a total turnaround time of only 29 days! (And remember, we weren't using internal mask and fab facilities at PARC, but were instead going to outside foundry services.)" (Conway 1982)

The results of this demonstration definitively convinced many in the community that this was an important shift. They were reported on at an M.I.T. VLSI conference in January, 1980. The technologies developed on the fly to enable assimilation of multiple designs eventually become MOSIS (Metal Oxide Semiconductor Implementation Service), a rapid turnaround prototyping service still in use today (see: http://www.mosis.com/). At Stanford, Jim Clark participated in a multi-chip project and created the Geometry Engine, the basis for the multi-billion dollar Silicon Valley icon Silicon Graphics. Numerous other companies were formed, many as "fabless" semiconductor companies which impacted the structure of the semiconductor industry itself (Baldwin and Clark 2000). The "Mead and Conway Revolution" had begun.

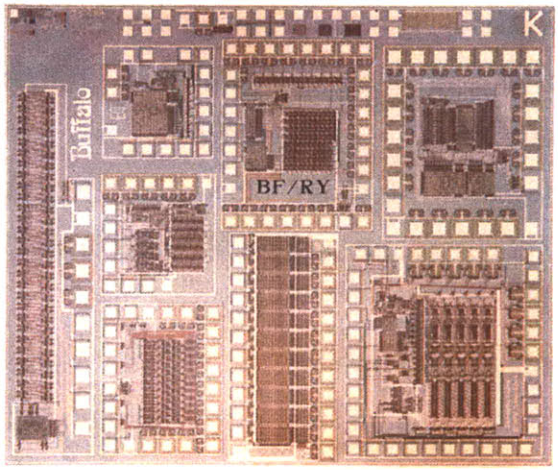


Figure 30: MPC79 Chip by Jim Clark at Stanford. The "Geometry Engine" which formed the basis for Silicon Graphics. From: http://ai.eecs.umich.edu/people/conway/VLSI

4.6 Analysis: Technological and Cultural Change

4.6.1 Summary of the Goals and Advances

The Mead and Conway methods were developed with the basic goal of simplifying design and liberating the design process from the tribalism and bureaucracy of contemporary semiconductor firms. As the analysis above demonstrates, this was accomplished through a combination of advances:

1. Invention of a new unit-less measure to describe design constraints (rules);
2. New software to check designs described with the lambda rules;
3. A new format (CIF) to exchange design files between designer and fab;
4. Reduction of the number of subsystem circuits for new designers to learn; and
5. Simplification/Removal of the separation between logic design and circuit layout.

The net result was a dramatic vertical decoupling of semiconductor design from fabrication, and the near simultaneous appearance of contract semiconductor foundries and fabless VLSI design companies. The industry remains permanently changed by this opening and continuous collaborative development of custom chips.

A number of lessons can be draw from these events and their impact. Within the context of open collaborative system development, two important lessons are most

chip designers, though implemented in the design courses and demonstration projects, never really took hold in the industry (Baldwin and Clark 2000).

Three basic factors created the conditions for the possibility of vertical decoupling in the industry. The first was the flexibility of manufacturing. The Planar Fabrication Process allowed any design to be fabricated on an existing fab by changing the masks. This meant that every new design could reuse the substantial investment in manufacturing technology already made. However, while necessary, this factor was certainly not sufficient to decouple the industry. Clearly, Mead and Conway did not invent the Planar Process, but they did use it to their advantage.

The second requirement for vertical decoupling was easy exchange of design information between design teams and fabs. For this Mead and Conway developed, promoted and gave away the CIF standard. The standard enabled any university teacher or fabless semiconductor company to send material in a way which could be decoded by various fabs. Yet, as described above, the need for a file interchange format between design firms and fabs was obvious, and widely used in industry at the time. CIF was developed not because no one had thought of decoupling the processes, but because no one wanted to share their proprietary format with Mead. The difference here was between an open and closed standard, not the invention of a standard per se.

The final requirement for effective vertical decoupling, given Mead and Conway's goal of democratizing design, was the development of a standard description of designs that could be run on any fab. For this, Mead and Conway created the lambda rules. This was a truly novel and important advance. Not only did the lambda rules abstract any given design from any given fab, but also in the process the new dimensionless unit abstracted changes in the knowledge needed from changes in the fabrication process. The lambda rules created a new language with which a community of designers could share and exchange ideas and designs. Significant

optimization and development of designs could occur without the need to coordinate with fab developers.

In examining these changes from a technical perspective, one finds three principles: (i) Simplicity (ii) Vertical decomposition using abstractions and (iii) Portability of design between vertical layers. Finally, we can also note that the design rules enabled a simplification of the design process via the removal of an important layer of abstraction. All of these changes were made in order to decrease the time to create a prototype.

4.6.4 Breaking the Culture of Secrecy

The development of the lambda rules, vertical decoupling of design and fabrication, and the simplification of the design process resulted in a tremendous explosion of economic activity. It is understandable that corporate managers would have been wary of some of these approaches. For example, sharing proprietary design rules could have decreased competitiveness. Even if spreading design knowledge ultimately broadened the market for semiconductors to the benefit of established players, these results could not have been predicted at the time. What is less obvious is why, if the Mead and Conway methods radically dropped the cost of prototyping, had they not been implemented internally and kept closed?

The simple answer to this question appears to be that the trade-offs required for rapid prototyping and system level design were seen as regressive to engineers and managers who had a different rank order of objectives. The methods improved the speed with which design could be accomplished and thus increased the potential volume of designs. Companies at the time, however, had grown to view the very things that inhibit such speed – deep expertise, extensive design teams, large divisions of labor, and highly precise designs – as their source of competitive advantage. The very structure and culture of the organizations were thus antagonistic to the Mead and Conway approach.

To take one example, concerns about liability caused fab engineers to oppose outright the very idea of rapid prototyping. Under existing rules, fab engineers were held accountable for the yield on every production run – meaning the number of usable designs versus non-working designs (Mead 2009). This meant that the engineers had to check each design before every run, which would not be worth the effort if the design itself was only a test. Fabrication operators rejected Mead's request as simply not a valuable exercise and a tremendous waste of money. As Mead later said, "people at Xerox's fab in El Segundo were very skeptical of what we were doing. They didn't like the idea. It was, as someone said, like a Protestant rite at a Catholic church." (Mead 2009) Rapidly prototyping was simply "not done here" according to the fab engineers, and would thus require higher-level buy-in to be implemented. If fabrication engineers within Xerox would not promote the simplifications required for rapid prototyping.

Design engineers also rejected the simplified methods because they did not address their current needs. Design engineers who were part of the "German army" described by Mead and were in fact valued precisely because they had specific expertise that others in the company did not have. A simplification of the design process would make them less relevant. Perhaps, due to this, many insisted that the simplifications gave up too much performance. As Mead notes, some "saw the simplifications as giving up performance. There was some truth to it. But what we saw that nobody else saw was that you gained a lot more by letting designers figure out system issues" (Mead 2009).

Conway discussed the reaction to their proposed methods in similar terms, "Our 'non-optimal' [lambda] rules later became very controversial, and caused considerable backlash from establishment figures who didn't seem to recognize the issues of computational complexity, teachability, and time-to-design tradeoffs that we were concentrating on - - designs created under these rules looked so "clean and simple" that they at first looked like "toys" to establishment folks who didn't understand what we were doing" (Conway 1999).

In hindsight it is clear that the focus on a new set of goals, including teachability, ease of design and cost of prototyping, ran contrary to the culture and organizational structure of established companies. Although the Planar Process enabled flexible manufacturing and, by extension, created the possibility for opening the design process, development of such an approach had been inhibited for some time due to the value structure within the companies.

This case, however, allows us to be more specific about the source of this cultural inertia. Not only were managers concerned with secrecy due to the need to protect proprietary information. But the very top level goals of the organizations, including the desire to create increasingly sophisticated systems, created a division of labor that encouraged increasing precision at the expense of rapid development. This resulted in a complex design process which, somewhat ironically, many companies viewed as a source of their competitive advantage. Given that managers viewed the expertise required by the 'German army' as an important asset, sub-system experts would have been highly valued. Simplifying and lowering the barriers to innovation via the use of the lambda rules and vertical decoupling not only threatened potential trade secrets, it threatened the value of knowledge held by formerly important members of the companies. For Mead, "One of the lessons here is how cultural changes in an industry are much slower than changes in technology or the potential of applying the technology." (Mead 2009)

4.7 Conclusion

4.7.1 High Level Observations about Opening VLSI

The story of opening VLSI in the late 1970s is fascinating for both technological and socio-cultural reasons. A number of important conclusions can be drawn from this case, both for OCSD development and for the design of complex integrated systems more generally. At the highest level, due weight should be paid to Mead's overall conclusion that the most important aspect of all of their endeavors was to remove

striking. The first involves the specific technical changes that were made, and how these facilitated the decoupling of the industry and the collaborative design process. It is clear that descriptive and vertical interface standards played the greatest role in this story and that horizontal standards were less important.

The second involves the socio-cultural story of how engineers and managers reacted to the proposed changes. Although they greatly dropped the cost of prototyping and eventually greatly expanded the market for VLSI systems, the changes also impacted precision in design and implicated core competitive aspects of the companies such as IBM and Xerox. In this way, the reaction to VLSI design can be seen as an example of technological lock-in, as described by Christensen in the Innovator's Dilemma (Christensen 1997). Firms operated within a value network that was contrary to that proposed by Mead and Conway. Both of these lessons are examined in turn.

4.6.2 Modularity Alone does Not Explain the Importance of the Methods

From the technological/economic perspective, a number of authors have suggested that Mead and Conway made the greatest advance in modularizing the design of VLSI circuits. Baldwin and Clark, for example, argue that the pair "totally re-conceptualized chip designs in terms of nested, regular, modular structures." (Baldwin and Clark 2000) They also suggest that, in total, the Mead and Conway methods were influential due to their task structure, their design rules, and their activities with respect to system integration (Baldwin and Clark 2000).

However, the data presented here suggests that modularizing design does not alone explain the economic explosion that followed the demonstration of their methods. To begin with, it is clear that the design process within companies was already highly modular. As noted, both Mead and Conway lamented the break down of design problems into a 'German army' of highly specialized sub-pieces, and their method in fact removed a layer of that decomposition. Conway has stressed that the idea of using effective problem decomposition was not new, writing, "The importance of effective problem decomposition for taming large search problems

has been recognized in AI for many years." (Conway 1982). Beyond modularity, design rules were also already widely used as heuristics for guiding design. As Mead demonstrated, "all of the fabs had their own design rules, and I was writing simpler design rules in the early 1970s." (Mead 2009)

Neither was the use of standards in the design of VLSI chips completely new. As noted in the sections above, the use of standard chips for LSI was common practice. Mead and Conway note in their textbook that library-based design of complex LSI circuits was already possible through the Polycell process (Mead and Conway 1980). While the Polycell process theoretically enabled rapid prototyping of custom chips, it failed, according to Mead and Conway, in that it gave up too much performance due to area power and increased delay times (Mead and Conway 1980).

Taken as a whole, the use of a structured design approach to complex problems and the creation of new design rules is not sufficient to explain the importance of the methods. As Mead said, "Splitting the design up is certainly a requirement for any complex system. There needs to be a lot of people working on the same page. I don't see that what we did actually made that very different in that respect....Now having computer tools to organize the design and check it and all those things makes that effort easier - but that's true in any case." (Mead 2009) The operative question is thus not whether Mead and Conway used these approaches in their novel process, but what exactly was new about the way in which they used these processes, and why this had not been implemented earlier?

4.6.3 Understanding What Was New

An answer to these questions requires deeper analysis of the kinds of standards developed by Mead and Conway, as well as the broader cultural context in which they operated. With respect to the former, it is clear that the vertical decoupling between design and fab was the critical event that enabled fabless semiconductor companies and foundries. The horizontal, modular, decoupling between teams of

what economists would call the transaction costs associated with developing new designs. Mead states:

"Liberating the design process from the bureaucracy was the biggest thing that happened...there is a bunch of work you have to do to get that to happen, but that was always the end goal for me. Now you could have innovation where you couldn't when it required management decisions to let someone go ahead and do a design. That is such a roadblock to innovation. So we wanted to lower the barrier to entry to enable smart people to innovate. The more innovation you have the better it will work." (Mead 2009)

This analysis fits well with the notion that there are costs to organizing internally or using the market that might be removed by employing an open, collaborative process. Further, it is critical to note that those within the bureaucracy were not easily convinced that opening would be useful. From their perspective it was very difficult to envision the indirect effects that spreading design knowledge would have on the semiconductor industry as whole.

The cultural resistance to democratizing design highlights the relationship between technological change and organizational change, which merits further investigation. Viewed as a whole, the question arises as to whether Mead and Conway made significant advances, or simply accelerated a decoupling in the industry that was inherent in the Planar Fabrication Process. Mead himself stressed that structured design, modularity, and computer simulation were already recognized as important contributions to the design of complex systems. And the design rules used by different fabrication facilities were quite similar. Was this a case of simply connecting the dots, and opening something that would have been forced open eventually? Or is it a deeper example of how sharing information can often create much greater gains than anticipated? Would the semiconductor industry remain closed without Mead and Conway's efforts? Here we enter the debate relating to technological determinism which, though fascinating, is outside the scope of this thesis. However, we also enter into the relationship between technology and organization, which highlights important sub-observations from the case.

It would appear, for example, that the case of opening VLSI design represents a "disruptive innovation", in the sense used by Christensen, in that a new value network with a new rank order of metrics was created. (Christensen 1997) It also suggests some ways in which this theory can be extended. In particular, there appears to be a kind of feedback loop between the value of knowledge within the organization and the value network within which the organization resides. More specifically, the rank order of goals pursued by the organization implies a division of labor that implicitly or explicitly values certain kinds of expertise. Once established, this division will be difficult to change precisely because those in roles suited to see how changes might be used may also find their position within the organization marginalized by the need for new kinds of knowledge. Further, the expertise within the company will often be viewed as a source of competitive advantage unless outside circumstances, such as the ascendence of a distruptive technology, force a re-evaluation. The feedback loop for valuing knowledge will be a force for stability in the rank order of goals pursued by the organization.

The question of expertise affects organizational dynamics particularly if the two value networks involve opening versus closing innovation. Open design by definition places value on outside knowledge, expertise, and skills. Yet, the benefits accruing from this reliance are indirect and difficult to quantify. In this context, it is clear that moving from closed to open innovation is very difficult and may require a completely new institution or organization. This surely merits further investigation.

Finally, Mead and Conway ultimately sought to reduce barriers to innovation by removing transaction costs and simplifying design. As with a number of examples cited in Chapter 3, this objective could theoretically be sought within a closed organization as well. The reuse of designs and decoupling of design knowledge could result in a faster pace of innovation even within a given company. This would require an internal investment in appropriate standards and design tools.

It seems, however, that the changes needed to increase "design volumes" are precisely those needed to open technology to third party developers. That is, once design elements are decoupled, and common design tools are shared within an organization, these same tools and decompositions can easily be shared more broadly. If a company would like to increase the pace of innovation radically, they may need to carry out the same activities that one would to open innovation *anyway*. If that is the case, the company may find that once the investment is made, opening actually provides a greater benefit than closing since it leverages network effects associated with large communities of designs. These observations require further investigation.

4.7.2 VLSI Design and the Proposed OCSD Framework

Opening VLSI design fits neatly within the framework and taxonomy developed in previous chapters. It is an example of the second strategy for opening in which a technology is fairly well understood, but all of the end applications are poorly developed. In this case, the end applications were poorly developed because there were simply so many of them.

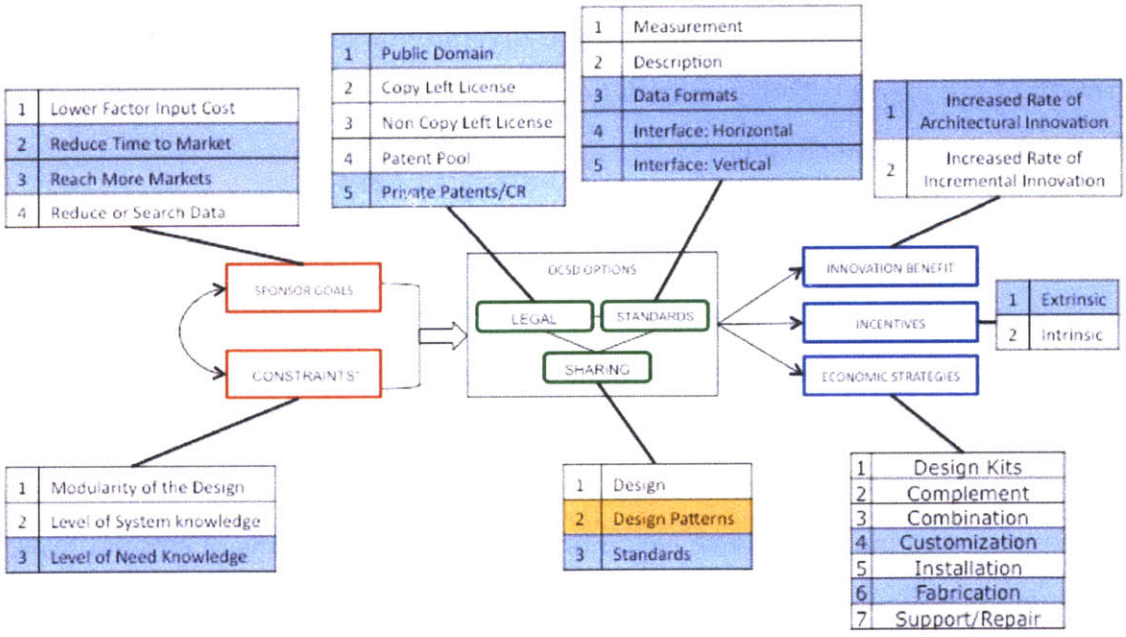


Figure 31: Framework with VLSI design elements highlighted in blue. Design Patters are highlighted in yellow because the sponsors of the regime originally tried to share these, but they were later abandoned.

Figure **31** depicts the framework with the important elements of the VLSI case highlighted. It demonstrates that the basic goal of the sponsors (Mead and Conway) was to lower barriers to innovation and thus to reduce time to markets and reach more markets with VLSI technology. Figure **31** further highlights that the primary legal mechanisms used were public domain standards and proprietary designs. Ultimately, VLSI circuits were protected under a new regime of MASK rules. Regardless, the methods had their greatest impact by opening standards and not designs. The use of shared design libraries never became common practice in the industry. As a result, the primary motivations fueling the development of the open model were extrinsic rather than intrinsic.

Initially, Mead and Conway promoted the use of both shared design information, in the form of circuit libraries, and shared standards. However, the former never took hold. The latter enabled a radical vertical decoupling of the industry. As a result, developers within this ecosystem pursued two basic economic strategies. The first involved selling fabrication services (foundries) and the second involved custom designing chips to be sent to these services (fabless semiconductor companies). The net outcome was a radical increase in the diversity of semiconductor designs and growth of the market. Today, fabless design accounts for approximately half of the market for integrated circuits.

In conclusion, the VLSI case validates the proposed framework and taxonomy. Further, the case validates one of the three strategic bundles identified in the Chapter 3. It suggests that vertical standards, made portable in this case by fab-independent design rules, may be very important if this model is to be pursued. This is very similar to the API model, described in Chapter 3. It further suggests that decoupling of an industry may have an impact even if the sharing of circuit libraries does not. The relationship of these conclusions to synthetic biology is explored in Chapter 7.

5 Applying the OCSD Framework to Microbial Fuel Cells

5.1 Introduction

The previous chapters developed and validated a framework for open, collaborative system development and posited three, high level strategies – Incremental Development, Database Development, and Architectural Development – that recur within the framework. How could the framework be used to for biological systems? This chapter uses a biotechnology called microbial fuel cells (MFCs) to describe how the *Incremental Development* and *Architectural Development* strategies might be applied within the biological engineering context.

Microbial fuel cells are chosen as a case study for a number of reasons. First, their range of applications and multidisciplinary nature enable rich examination of the relationship between biological design and traditional engineering. Second, genetic engineering of microbes with exo-electrogenic capabilities such as *Shewennalla oneidensis* has been carried out within the context of the iGEM competition. Therefore, preliminary approaches to library-based design of genetic circuits, as advocated by many synthetic biologists, can be explored within this case (Noll 2006; Fredrickson, Romine et al. 2008; Lovley 2008). The case study is somewhat limited by the fact that MFCs are still an emerging technology and have not yet been commercialized. However, most biotechnologies, besides some pharmaceuticals, are immature. Further, previous chapters demonstrated that high levels of technological uncertainty could be a motivation for, rather than a barrier to, OCSD.

As a whole the chapter clarifies how the framework and strategies can be used to design an OCSD in the bioengineering context. Biological-specific benefits and costs

are defined and analyzed, and potential practical solutions proposed. The design exercise illustrates how the framework can be used to guide development. In doing so, it also raises important questions that should be addressed through future research.

The chapter is organized as follows: section two introduces microbial fuel cell technology, including potential applications and critical design challenges. Section three introduces how the framework can be used to guide the development of an open platform for MFC development. Section four describes how the Incremental Development strategy might be applied for microbial fuel cells treating wastewater. Section five describes how the Architectural Innovation strategy might be applied for biosensors built around a microbial fuel cell platform. Section six is a conclusion.

5.2 Microbial Fuel Cell Technology

5.2.1 Technology Background

Microbial fuel cells (MFCs) are a class of fuel cells in which the catalyzing agents on electrodes are microbial bio-films rather than noble metals such as platinum. The "fuel" can be a diverse array of organic material instead of chemicals such as hydrogen or methanol.[15] MFCs function because microbial species such as *Geobacter sulfurreducens* and *Shewanella oneidensis*, among others, have the remarkable ability to respirate directly to metals in their environment. In effect, microbes at an anode use organic matter as food and metals as terminal electron acceptors in their respiratory chain (Logan 2008). Reverse reactions occur at the cathode with oxygen or other oxidized species serving as the electron acceptor.

As with traditional fuel cells, oxidation of anodic material is accompanied by the production of ionized species which are transported to the adjacent chamber (see

[15] Microbial Fuel Cells actually can use hydrogen or methanol as fuel, since these can be metabolized by the microbial catalysts. But they need not use these.

Figure 32). An electrolyte can be used to selectively allow charged species to move between the anode and the cathode.

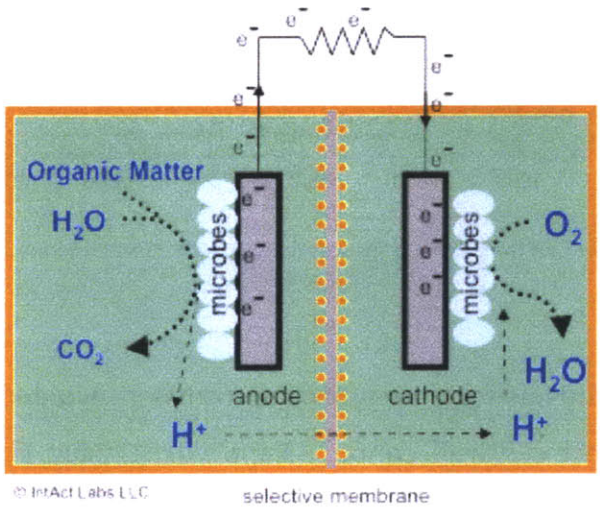


Figure 32: Schematic of a Microbial Fuel Cell functioning with bio-anode and a bio-cathode.

Like most good ideas, MFCs have a longer history than many assume. The basic phenomenon of microbial exo-cellular electron transport to electrodes was identified in the early 20th century (Potter 1911). Since then, significant work has gone into understanding underlying mechanisms. In the 1960 MFCs went through a resurgence with funding from NASA due to their potential to clean waste while generating electricity (Bennetto 1984). However, it is only recently that researchers have developed MFC devices with high power levels utilizing *direct* electron transport between bacteria and metals (Kim 1999). Figure 33 below illustrates the near-exponential rate of innovation in the field since the discovery of direct-electron transport in 1999. These higher transport rates result in significant increased electricity production, opening the door to a broad range of applications.

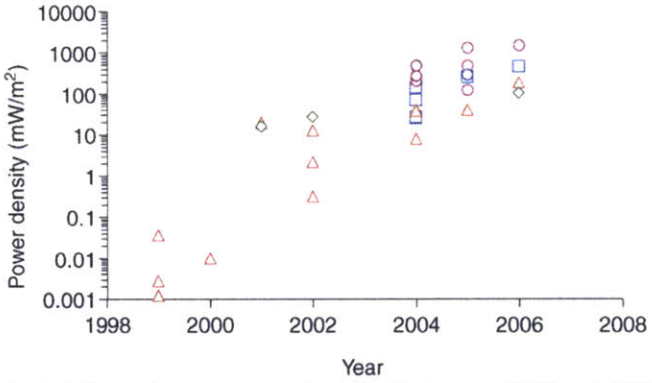


Figure 33: Microbial Fuel Cell maximum power densities between 1999 and 2006. (From: Logan and Regan, 2006)

Two plausible mechanisms for direct electron transfer included so-called microbial nano wires and direct cell-wall contact. Nano wires (see Figure 34 below) are electrically conductive pili that microbes use to attached directly to metals and to conduct electrons (Ntarlagiannis, Atekwana et al. 2007). Figure 34 below shows electrically conductive microbial nano wires produced by a bio-film of Geobacter growing on an electrode.

Figure 34: Geobacter biofilm with electrically conductive pili. (courtesy: New Scientist)

Direct contact of microbes with electrodes is another potential mechanism for high-rate current production (Logan 2008). This mechanism seems to be enabled by outer-membrane c-type cytochromes known as OmcB and OmcA (Fredrickson, Romine et al. 2008). More recently, MFC designs have been developed that produce high power outputs with indirect electron shuttles, rather than direct contact (Ringeisen, Henderson et al. 2006). While it is not yet clear which of these three mechanisms predominates for a given species or microbial community, the higher rates of electron transfer can nevertheless be used in a range of applications.

5.2.2 Applications

As a kind of fuel cell, there is necessarily interest in using MFCs to generate renewable electricity on a large scale. However, Figure 33 above makes clear that despite a near exponential rate of innovation, power densities remain low compared to traditional fuel cells and batteries. Maximum power densities of up to about 4 W/m^2 have been reported in the lab (Logan 2008). Real world power outputs drop with larger electrodes, impurities in the input media (e.g. fuel), and other sources of losses such as increased internal resistance, contact resistance, and columbic losses (Keller and Rabaey 2008). Even at densities of 4 W/m^2 a typical home, consuming an average of 1 KW of electricity, would require more than 250 m^2 (2,690 ft^2) of electrode surface area. Though not outside of the realm of possibility, these estimates suggest that MFCs may not be practical for large-scale renewable energy production in the near future. However, another breakthrough could change this.

Besides electricity production, MFCs can be used to treat water in an energy efficient manner. For this application, the oxidation of anodic material removes organic pollutants such as dissolved Carbon, Nitrogen, or Suspended Solids while simultaneously generating some electrical current. By contrast, current methods to remove pollutants from wastewater, such as aerobic respiration, are exceedingly energy intensive, consuming over 3% of electricity across the United States (Logan 2008). In 2004 Liu and colleagues definitively showed that a single-chamber, flow-through, microbial fuel cell (see Figure 35 below) could treat real world domestic wastewater in the anodic compartment while generating rather than consuming electricity. (Liu, Ramnarayanan et al. 2004) Their reactor produced a maximum of 26 mW/m^2 while removing 80% of the chemical oxygen demand (COD – a measure of dissolved carbon).

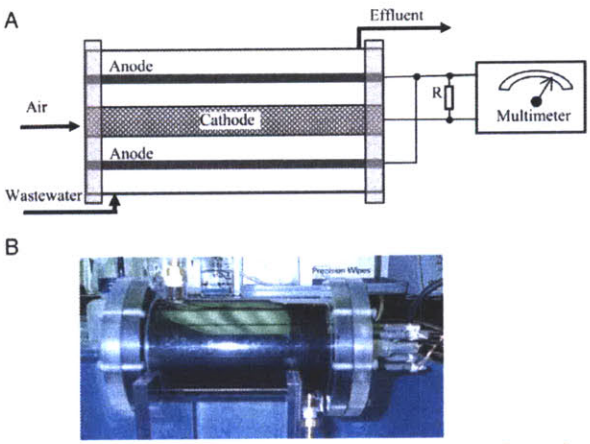


Figure 35: Schematic (A) and picture (B) and the first single-chamber, flow-through MFC reactor used to treat domestic wastewater. (From: (Lie, Ramnarayanan et al., 2004))

Since this report a number of single chamber and dual chamber MFCs treating a variety of wastewaters have been developed (He and Angenent 2005; Logan 2005; Oh and Logan 2005; Aelterman, Rabaey et al. 2006; Keller and Rabaey 2008). Recently, a large-scale MFC pilot plant (1000 liters) built at Fosters Brewery near Brisbane, Australia successfully remediated brewery wastewater while generating approximately 8 W/m^3 of volume (Figure 39) (Keller and Rabaey 2008)

The bio-electrochemical processes powering MFCs have shown great flexibility in the basic application of treating water while generating value-added products. Beyond electricity production, the electrons liberated by anodic oxidation have been used to synthesize a variety of chemicals at the cathode including hydrogen (Logan and Grot 2006), methane (Clauwaert and Verstraete 2009), and hydrogen-peroxide (Rozendal, Leone et al. 2009). Further, biological cathodic processes, which reduce rather than oxidize substrates, can be used to remove oxidized pollutants such as nitrates (Virdis, Rabaey et al. 2008).

Both electricity production and wastewater treatment depend on achieving high rates of electron transport. However, MFCs have a diverse array of low-current applications. One promising class of such applications involves powering distributed sensor networks (Reimers, Tender et al. 2000). Another class of applications would exploit the small size of microbes to generate micro-MFCs with the capability to

power devices at scales well below existing batteries (Chiao, Lam et al. 2006; Ringeisen, Henderson et al. 2006). Miniature MFCs could power medical devices (Chiao, Lin et al. 2007) or be used as micro-scale air and water sensors (Noll 2006). And some have looked into the possibility of using MFCs as a way to enable robots to derive power from the digestion of real food (Wilkinson 2000).

In short, as a new interface between biology and electricity/electronics functioning effectively across a range of scales, exocellular electron transport from microbial biofilms promises a diverse range of exciting applications. The breadth of applications suggests that the field will likely spawn sub-disciplines targeting end-goals with highly different constraints and integration needs. These sub-domains will be supported by continued scientific research into the basic bio-electrochemical processes. Specific design challenges will then depend in part on end applications.

5.2.3 Generic Design Considerations

While objectives and constraints will vary based on end applications, we can frame the MFC design problem with application to water treatment, adding to or removing items for other application areas. At a high level, the objective of MFC design for wastewater treatment is the simultaneous maximization of the energy density, P_D (Watts/m^3) and BOD removal rate, Br (KG/m^3Day) at minimum cost, C ($/$m^3$). This constitutes a multi-objective optimization problem which could be solved using scalar or Pareto methods (De Weck 2004). These three objectives are defined by overlapping design variables and parameters affecting the biology, electrochemistry and chemistry of the input fuel. The three critical "modules" that must be considered are therefore the reactor, R, the biofilm, B, and the Fuel or Input, F (Figure 36).

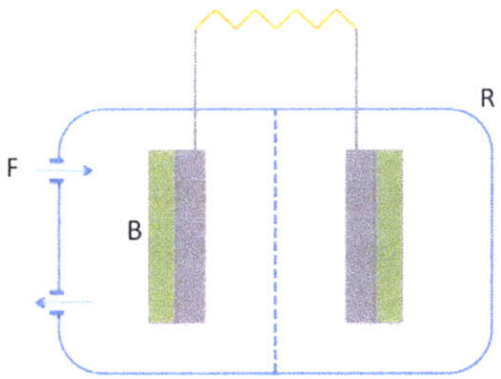


Figure 36: Schematic of a Microbial Fuel Cell. Main elements include: 1. The fuel, F, which can be organic matter including wastewater. 2. The Biofilm, B, which serves as a catalyst and can be used on either the anode (left) or cathode (right). 3. The reactor, R, which includes the casing, electrodes, electrical connections, and other non-biological items for moving fuel.

From an electrochemical standpoint, the critical design concern is to avoid over-potentials and other efficiency losses which increase as current is drawn (Clauwaert, Aelterman et al. 2008). The next chapter in this thesis summarizes the basic losses associated with the design of any fuel cell in more detail. For the present purposes we note that maximizing power production involves minimizing voltage losses while drawing current from the cell.

For an MFC treating wastewater, two critical additions must be made to this electrochemical picture. First, the electrochemistry at the anode or cathode (depending on the design) is governed by biological phenomena. Second, the consumption of fuel (organic pollutants) is to be encouraged rather than minimized. Maximizing fuel consumption involves designing or encouraging the growth of bio-films with elevated metabolic activity. This may or may not translate to higher power in an MFC, depending on the ratio of electrons used for cell growth versus other functions.

Beyond increasing catalytic activity and fuel consumption, the biofilm at the anode and cathode will be heavily affected in its ability to oxidize fuel by the nature of the surrounding environment. The electrode material and structure of the biofilm itself affects the build-up of waste products such as CO_2 and the transport of protons, ions and cations, to and from the surface of the electrode (Picioreanu, Head et al. 2007;

Clauwaert, Aelterman et al. 2008). These can put the need for elevated current levels at odds with the need for cells to grow in moderate pH conditions. Finally, the ability of a biofilm to maximize use of the organic material for growth is also affected by the nature of the input fuel. Pre-treatment of the input fuel can help in this regard.

5.2.4 Critical Unknowns and Challenges for Real World Applications

The previous section discussed the major factors for designing MFCs and presented an idealized framing of the design problem. Yet, important fundamental uncertainties exist which remove design control and limit power outputs. The low level of maturity in the field of both MFCs and bioengineering more broadly creates fundamental scientific and engineering uncertainties that make designing reactors, let alone biofilms or bacteria, a highly iterative and uncertain process. In order to apply the framework for open design to the field, it is important to clarify these scientific and engineering questions.

First, the basic processes associated with exo-cellular electron transport are poorly understood. While much electrochemical work has gone into characterizing MFC performance with different reactor architectures, materials, and fuels, most of these studies do not thoroughly analyze the biological phenomena (Lovley 2008). It is still not clear how the three electron-transport mechanisms described above – nano wires, direct cell-wall contact, and indirect electron shuttles – are activated and controlled within a cell's regulatory network. For example, experiments have not yet determined whether exo-cellular electron transport is affected by surrounding cell densities, metal composition, pH, or other environmental and cellular issues (Lovley 2008). Despite our lack of knowledge of the underlying biology, some steps have been taken to tweak single strains for increased respiration rates (Izallalen, Mahadevan et al. 2008). While promising, these are far from full rational design of biofilms, or even improved single bacterial strains.

Three primary sources of reduced power in microbial fuel cells are activation,

concentration, and ohmic losses. Decreasing activation and concentration losses requires developing strategies for mitigating the affects of pH and ion gradients and other factors at the surface of electrodes. It appears that the build-up of pH within anodic biofilms can greatly hinder operation (Picioreanu, Head et al. 2007; Keller and Rabaey 2008). pH gradients across membranes also creates voltage losses (Clauwaert, Aelterman et al. 2008). These problems could be overcome with changes to reactor geometry, electrode geometry, or biofilm structure and pH tolerance. Clarification of the relationship between pH and biofilm exo-cellular electron transport, would enable targeted solutions based on either biological engineering or reactor engineering.

Ohmic or internal resistance losses are caused largely by the nature of the electrolyte, the choice of materials, and the shape of the fuel cell itself. To date, most microbial fuel cells have exhibited high internal resistance and low columbic efficiency (Logan 2008). One way to reduce ohmic resistance is to reduce electrode spacing. However, this can have the negative affect of increased pH and runs into the problem of concentration losses described above (Logan 2008). While these relationships between ohmic, activation, and concentration losses and biofilm pH tolerance have been identified, there is no clear model to describe their interaction. This limits the efficiency of a rational search for new reactors and microbes.

Low columbic efficiency, another source of losses, can also arise from cross-over of oxygen between anode and cathodic chambers, and from incomplete oxidation of fuels (Noll 2006). The former can be limited by a dielectric membrane, but this can create pH gradients and increase internal resistance. Incomplete oxidation of fuels can be mitigated by longer hydraulic retention times, and other strategies.

Besides maximizing power and efficiency, a number of challenges are specific to other applications of MFCs. For example, if used to treat water, the diversity of input fuel can hinder standardization of MFC designs. Variations in wastewater composition can affect all aspects of fuel cell performance, by altering the food-

source for microbial communities and by altering the internal resistance of the electrolyte, depending on the cell design. More generally the robustness of the microbial biofilm must be considered. Fluctuations in pH, temperature, substrate type, shear strength of moving liquids, and other factors can all impact the long-term viability of a given design and application.

Finally, as a highly multidisciplinary field encompassing microbial biology, electro-chemistry, materials science, electrical engineering, and domain-specific fields such as energy and water treatment, MFC advances are hindered by conflicting methods for describing phenomena. Standard measures and units are still being developed, and even simple metrics such as columbic efficiency may be calculated in very different ways. This challenge, which bears on the standardization of measurement and description, makes comparison of results challenging.

5.3 Open, Collaborative Design of MFCs

5.3.1 Applying the Framework and Strategies

How might one contemplate using an OCSD process to guide development in the field? The proposed OCSD framework described in previous chapters includes three strategies: Incremental Development, Database Development, and Architectural Development (Table 11).

Table 11: Three OCSD Strategies.

	Name	Typical Sponsors	System Knowledge	Market/Need Knowledge	Principal Elements Shared	Important Standards	Type of Innovation
1	**Incremental Development**	Users	Low	High	Entire Designs	Measurement, Description	Incremental
1.A	**Database Development**	Users or Developers	Low	High	Data	Measurement, Data Formats	Incremental Pre-Competitive Research
2	**Architectural Development**	Developers	High	Low	Design Patterns, Interfaces	Vertical Standards, Data Formats	Architectural

Each strategy is defined by the type of innovation sought by the OCSD sponsors, and

provides suggested constraints on the "inputs," "design options" and "results" or objectives (see Figure 37). Inputs, design options, and objectives not constrained within the strategy are free to be altered as needed.

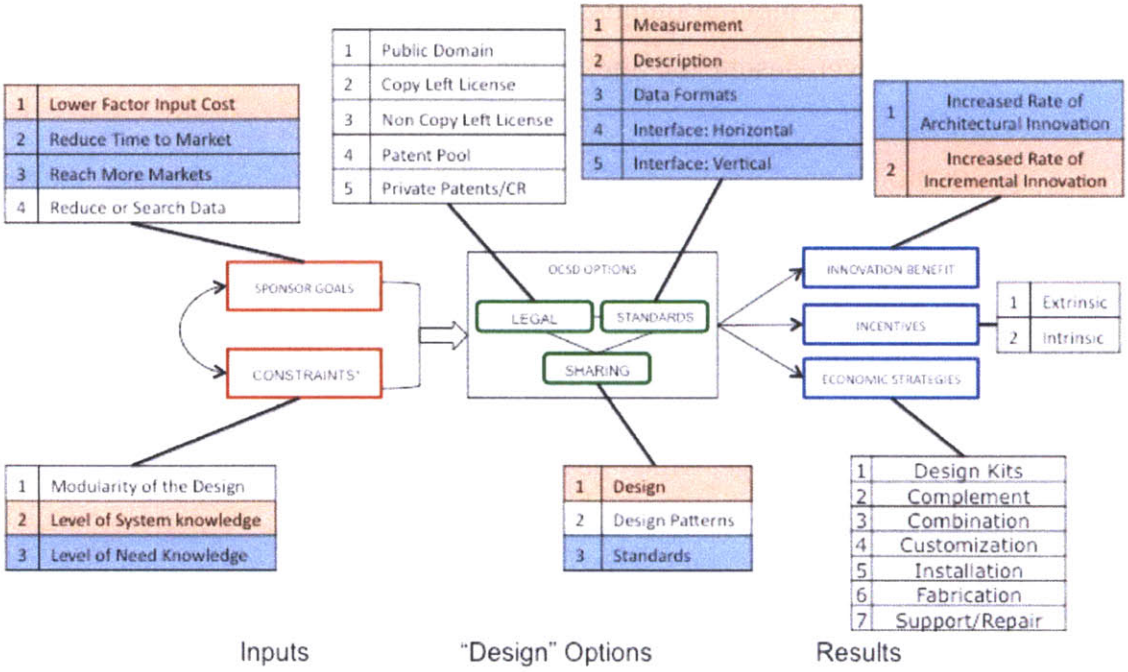


Figure 37: Incremental Development Strategy (orange) and Architectural Development Strategy (blue) are presented in the context of the broader framework.

In order to employ the framework and a given strategy, then, one must first define the potential sponsor of the regime and the kind of knowledge that it has about the market and the system being development. This then places constraints on the potential end-goal for the OCSD process, and suggests constraints on design options including what should be shared and what kinds of standards should be developed. For example, in cases where a sponsor has low system knowledge due to integrality, they should share an entire system architecture with a very defined end-goal. In this case, it will be important to describe the architecture consistently, and determine what is open/standardized. In contrast, where a sponsor has high system knowledge, but low knowledge of potential applications, it is suggested that they share interface standards and data-formats in order to increase the range of potential end applications. In this approach, as described in previous chapters, one must be careful to distinguish interfaces *within* a given layer in the value chain, and those *between* layers. As the VLSI chapter made clear, standard interfaces between

layers is a more fundamentally powerful method for opening systems, than opening within a layer.

Given the high level of the options within the taxonomic categories, there will remain a range of decisions to make within the constraints suggested by the strategies. Therefore, additional tools might be used to craft specific strategies, as discussed below.

5.4 Applying the Incremental Development Strategy

5.4.1 Summary and Assumptions

Figure 38 below presents this Incremental Development strategy in the context of the wider OCSD framework. The strategy involves choosing a fixed market application and a sharing an entire system architecture around which sub-systems can be developed, in an effort to collectively improve performance through incremental advances by various developers.

The strategy was found to be employed more often by users of a technology, as a way to decrease costs associated with acquiring or operating the system. For example, in a case described in previous chapters (Cleveland), owners of iron mines shared information on blast furnace designs in order to more rapidly develop efficient designs, which reduced their fuel costs. The fixed end-use, in this case, involved processing of iron ore into pig iron.

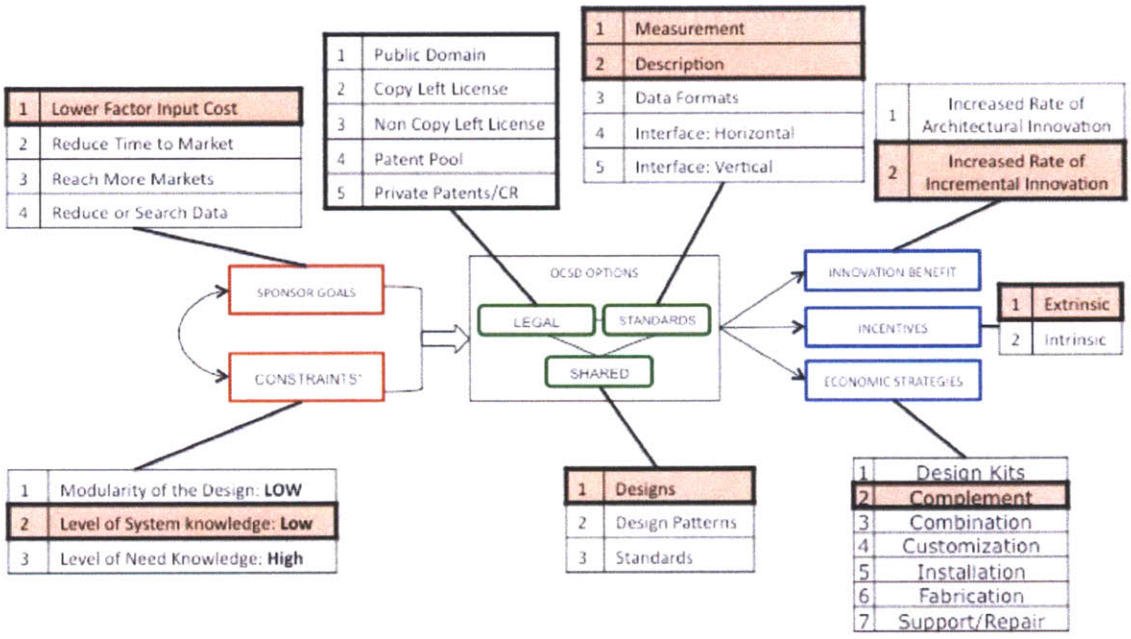


Figure 38: Constraints and decisions within the Incremental Development strategy.

This section, describes how each set of options can be operationalized for the commercialization of MFCs. It then analyzes benefits, costs, and opportunities associated with applying the strategy. As discussed in more detail below, in order to describe the use of this strategy, one specific market-application must be chosen. In this case the application chosen is wastewater treatment.

5.4.2 OCSD Inputs: Sponsor Goals, Constraints

To apply this strategy, we can use the fixed goal of wastewater treatment. The function, in this case, it to remove pollutants, such as dissolved Carbon, Nitrogen, and Phosphorous from water. To simplify the problem let us take the example of MFCs treating dissolved Carbon, measured in concentrations (mg/l) of biological oxygen demand (BOD), in the brewery industry. This section proposes a strategy and standards based on this approach. Sample requirements for a site would be to reduce the Biological Oxygen Demand (BOD) of water exiting a pre-treatment facility from 10,000 mg/l to 200 mg/l (EPA Compliance). A mid-scale brewery might produce 100,000 gallons/day (~ 378,000 liters/day).

5.4.3 OCSD Design Variables: Sharing Designs, Standards, and Legal Mechanisms

5.4.3.1 Shared Reactor Architecture

This strategy proposes that sponsors share the full design of a system architecture. The precise architecture is not critical to this discussion. For the purpose clarifying what is meant by reactor architecture, let us assume a reactor design roughly similar to a pilot MFC plant created to treat brewery water in Brisbane, Australia (Figure 39). The process had an up-flow configuration – brewery water enters the bottom of cylindrical tubes, flows along an inner bio-anode chamber, over the top, and onto an outer bio-cathode chamber. The anode removes BOD while increasing the hydrogen ionic concentration (pH) as the water flows up the reactor. Open-air biofilm electrodes catalyze oxygen reduction at the cathode.

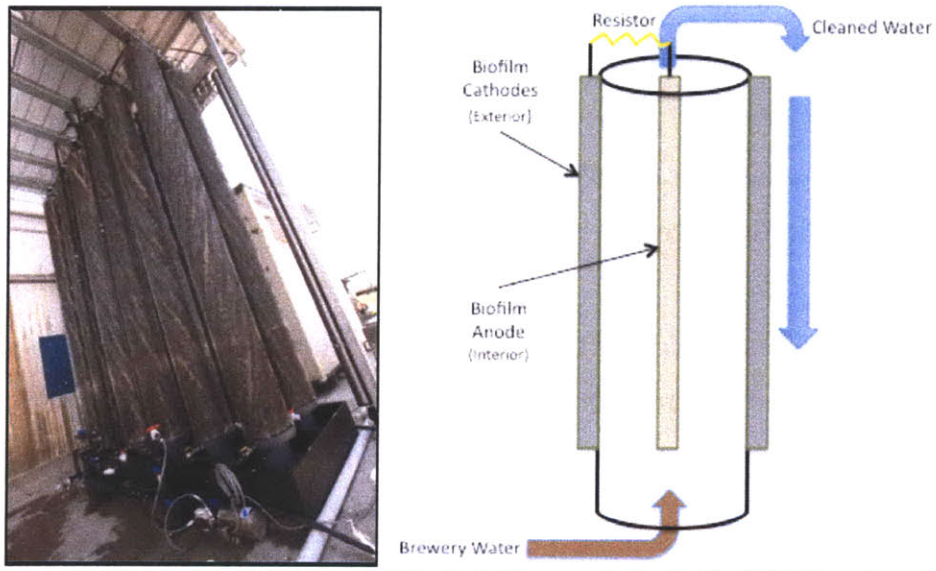


Figure 39: Left: 1000 liter MFC Pilot built at Foster's Brewery in Australia, 2007 (courtesy: Keller and Rabaey, 2008) Right: Schematic of the reactor architecture (created by author based on picture).

Critical variables within this architecture therefore include the width of the chamber, the distance between anode and cathode, the shape of electrodes, the number and kind of contacts between electrical components. The effect of these parameters will depend on the nature of the biofilm, as discussed below. Other aspects of operation can also be modulated, such as the water flow rate, the operating current, the input and output fuel concentrations.

5.4.3.2 Standards Definition and Analysis

The strategy indicates that internal interface standards are less important for this model because the entire architecture is in a sense both integral and standard. Important standards for the open regime would therefore include description of the architecture (for replication) and, depending on the exact goals, the mechanism for exchanging performance results and the interface standards for upgrading some components. The sponsors' goal would be to create a standard architecture and description to be shared among developers.

The strategy does, however, encourage the development of proprietary portions of the system architecture by developers wishing to sell complementary products. From a standards design perspective, then, there are two critical questions: First, what should be the values for fixed variables in the shared design? Second, what should be opened and what might be kept closed and proprietary? Answers to both of these questions depend in part on the nature of the technology and constraints associated with the biological medium. This section examines these questions at a technical, but qualitative level.[16]

Based on the brief description of the reactor architecture above, it is clear that the fixed elements of the reactor should be chosen with consideration for how the reactor and the microbes interact. Based in part on the discussion of design objectives and constraints above, we can identify a number of ways in which performance or elements of the microbes will impact and constrain the performance and operation of reactors and vice-versa. These are enumerated in Table 12 below.

Table 12: Impact of microbes on reactors and vice versa in a Microbial Fuel Cell.

Impact of Reactor on Microbes	Impact of Microbes on reactors
Electrode spacing affects diffusion between anode and cathode and thus Ph concentrations	Respiration rate as function of input affects HRT and required volume
Electrode spacing affects rate of diffusion of charged species	Respiration rate affects Ph build-up and therefore internal resistance

[16] A quantitative method for answering these questions is presented in chapter 6.

Surface of electrodes affects biofilm build-up	Ph tolerance affects maximum potential current and therefore
Geometry of electrodes affects diffusion of nutrients to biofilm	Biofilm structure impacts diffusion of wastes, Ph, other elements
Specific area of electrodes affects voltage and current requirements and impacts	Type of current transport affects potential hydraulics (batch versus flow-through
Membrane existence and type affects Ph gradients	Type of current transport affects set-point voltages between electrodes if applicable
Operation at fixed voltage affects biofilm efficiency and growth	Rate of current transport affects required electrode size
Size of reactor affects required hydraulic retention time	Biofilm robustness limits hydraulics - max sheer strength
	Biofilm metabolic diversity affects range of end-applications

Though not necessarily exhaustive, the list suggests recurring themes. First, an important reciprocal relationship involves the production and movement of hydrogen ions and other oxidized species. Parameters in the reactor affect diffusion of these species, creating variations in pH that can affect biological performance. Conversely, the catalytic performance of the biofilm and its structure affect how many ions and charged species are released, which can affect voltage losses by creating differential gradients. If either the reactor or the microbes are fixed, increasing current rate needs to be accompanied by consideration of resulting ohmic losses and pH tolerance respectively.

The relationship between electrode surface properties and biofilm also appears on both sides of the table. The nature of the surface will impact biofilm formation and structure. But biofilm structure and formation can change the requirements for electrode size, specific area, and other elements.

Interestingly, a number of parameters are complementary rather than in competition. These relate to the overall reactor design and performance, rather than electrodes. For example, advances could lead bio-films to support higher sheer strengths caused by water moving in the reactor. While sheer strength is affected by

reactor design, it could conceivably be changed once a reactor has been built simply by modulating the operating characteristics of the reactor.

The analysis suggests that, among the different reactor characteristics, it would be most critical to publish data on reactor chamber size and electrode spacing. Conversely, it would benefit users of the technology to develop reactor architectures for which the electrode spacing could be variable, or the entire electrode-biofilm stack could be insert and removed.

One way to accomplish this using the proposed architecture would be for biofilm designers to create tube-anodes of varying diameters, which could be inserted into the up-flow reactors (Figure 40). In this way, an "upgrade" would involve a new anode-biofilm coupling.

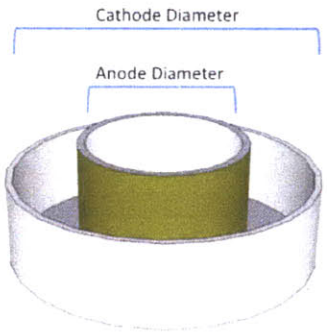


Figure 40: Suggestion for varying or fixing electrode distance based on a variable inner anode diameter, as a way to decouple pH and concentration concerns from current rate, catalytic rate and microbial tolerances.

Enabling variable electrode spacing is one of numerous potential ways of decoupling parameters with the MFC. Chapter 6 provides a quantitative method to explore the trade-offs associated with standardizing different elements of the reactor.

5.4.3.3 Legal Regime

The discussion above indicates some of the parameters and performance indicators that would be useful to disclose.

Parameters:

1. Reactor tube geometry: Height, Diameter
2. Total reactor volume
3. Input BOD concentration
4. Input Rate (Volume/Day)

Performance:

5. pH profile along each reactor tube
6. Power Density & Columbic Efficiency
7. BOD Removal Rate & Hydraulic Retention Time

As proposed within the broader OCSD framework, sponsors have discretion to choose any of the five legal mechanisms found to recur within OCSD more generally (see list in Figure 38 above). Further, it seems likely that, based on the potential decoupling described above, developers in an OCSD for MFCs treating wastewater will develop proprietary strains of bacteria that might "upgrade" existing systems. To encourage this, sponsors should take into account the interface between reactor and microbes, and ensure that there are no potential restrictions on patenting microbes.

These goals could be achieved by creating a license for use of the proposed architecture that does not necessarily require royalties, but does specify what kind of information is revealed about the resulting design and operation of the system. One could imagine, for example, copying the Lean Engine Reporter model from chapter 3, and requiring all firms using the architecture to post a distinct set of design parameters and performance results to a common Internet site. Because this would be enshrined in a user-license, failure to do so could result in substantial fees or some other penalty.

An open license and publication requirements would not mean that sub-systems

could not be patented and sold. In fact, it would give sub-system designers a method for promoting their inventions. Further, it would enable contractors or consultants to identify and incorporate well performing sub-systems more quickly.

5.4.4 OCSD Outputs: Incentives, Innovation, and Economic Strategies

5.4.4.1 Developer Strategies

The cross-case analysis in chapter three demonstrated that a wide range of economic strategies are employed by developers in any given OCSD regime. These include selling services such as fabrication or repair, and complementary products such as electrodes, microbes, or control systems. If the reactor itself is open, an important activity may be the sales of proprietary microbes to multiple deployed reactors.

This approach would like result in two waves of innovation. In the first, user-firms would contract for the construction of a number of reactors. As the market began to saturate, there would likely be a second wave in which outside firms would sell components, such as upgraded microbes or microbial communities to the existing infrastructure.[17]

5.4.5 Benefit and Costs of Applying the Incremental Development Strategy

We can separate benefit and costs of such a strategy for users and for developers. For users, the benefits are quite clear – lower-cost acquisition of a novel and higher-performing wastewater treatment system. This results in a shorter payback period. The costs may theoretically include some uncertainty that contractors have the right expertise, but this could be mitigated in various ways. From a user standpoint the principal concern, it would seem, is the possibility that developers have margins that are too low to support viable R&D for continued improvement.

[17] This wave might be characterized by economics much closer to those typically associated with information rather than physical goods. That is, because microbes are expensive to develop but cheap to replicate, the firms selling in the second wave would face a cost structure of high cap-ex and low manufacturing/distribution costs.

From the developers' perspective the benefits and costs are mixed. There are really two kinds of developers in this situation. The first is the company or individual who developed the architecture, at expense. The second are developers who might build on the architecture once it is opened.

For all users, a freely available architecture will both enable faster innovation around the core architecture and more rapid market adoption. More rapid improvement caused by a greater number of developers will, of course, increase competition and lower margins for all firms. However, it will also speed up deployment of the technology across the world. Because microbial fuel cells treating wastewater represent a high capital expense for companies, cost will be a critical determinant of the decision to acquire. Once acquired, that architecture will likely become locked-in. Rapid deployment therefore *creates a market* for sub-system developers and improves the chances for the MFC industry as a whole versus competing industrial technologies such as membrane bio-film reactors or novel anaerobic digesters.

Once a broader set of architectures is deployed, this might in fact raise margins for sub-systems, since the technology will already be locked-in. This depends on both the legal regime and ability to plug sub-systems into the existing architecture. Both of these factors are discussed below.

Because this strategy is employed when technological uncertainty is high, one of the benefits should be to reduce critical uncertainties in the scale and operations of such systems. This requires a mechanism to ensure that at least some kinds of information are shared among users of the technology. And this, in turn, requires that developers understand the exact kind of uncertainties that they would like to reduce.

High technological uncertainty suggests that some aspects are poorly codified and therefore trade secrets play a greater role in the design and operation of the

technology. A company that has developed the architecture will have more trade secrets – which will rise in value as the technology is increasingly deployed.

The difficulty for developers, especially those that have invested in developing the architecture, is that many of these benefits – faster adoption, a market for sub-systems and services, reduced uncertainty – are secondary. They tend to benefit the use of the technology generally, in comparison to competing technologies such as anaerobic digestion. But these industry benefits may or may not trickle back to the firm that opened the system. Further, the costs of elevated competition are particularly damaging to small firms that might be pushed out of the market by very large firms with other competitive advantages besides intellectual property.

However, some of the benefits might be mimicked in a closed regime. Developers could speed up deployment of proprietary systems by keeping margins on the reactor very low, with the goal of selling components later (the razor and blade model). Or, innovators might license the technology to numerous third parties who can speed up deployment. These benefits and costs, summarized in the table below, indicated the complexity of deciding to open an architecture in this case.

Table 13: Some benefits and costs of participating in an Incremental Development OCSD for MFCs for developers.

	Developer Benefit	Developer Cost
1	Lower transaction costs	More competition
2	More innovation, faster improvement	Lower margins
3	Faster adoption by users	
4	Market for components	
5	Reduced technological uncertainty	
6	Better economics versus competing technologies	

The benefits clearly fall within clusters. The first two relate to *innovation rate*. Three and four relate to the *adoption/diffusion rate*. Benefits five and six relate to benefits associated with acquiring knowledge and information from third parties. In essence, then, a developer utilizing an open strategy in this context must weigh benefits of volume and information against the costs of heightened competition. Further, in

order to ensure that the benefits actually accrue, the designer of such a regime would need to ensure that the architecture can actually be continuously improved, and that the correct information is shared. This is a function of standards and the legal regime, respectively.

5.5 Applying the Architectural Development Strategy

5.5.1 Summary and Assumptions

In contrast to the Incremental Development strategy, the *Architectural Development* strategy can be employed when knowledge of the system is higher but knowledge of end-applications is lower. In this case functional modules are shared along with a stable or integral base, to create diverse end-applications. Depending on the sponsor of the regime, this stable base may be proprietary (e.g. the API model) or it may be open shared (e.g. the GNU/Linux model).

While there remains considerable uncertainty both with respect to the engineering of reactors and microbes, we can make a few assumptions in order to describe how such a strategy might be employed. In particular, there has been substantial work to validate this model of biological engineering through efforts such as the Registry of Standardized Biological Parts and the iGEM Competition at MIT. This section therefore outlines this approach using results from recent iGEM projects, and discusses the benefits and costs in the context of a broader OCSD.

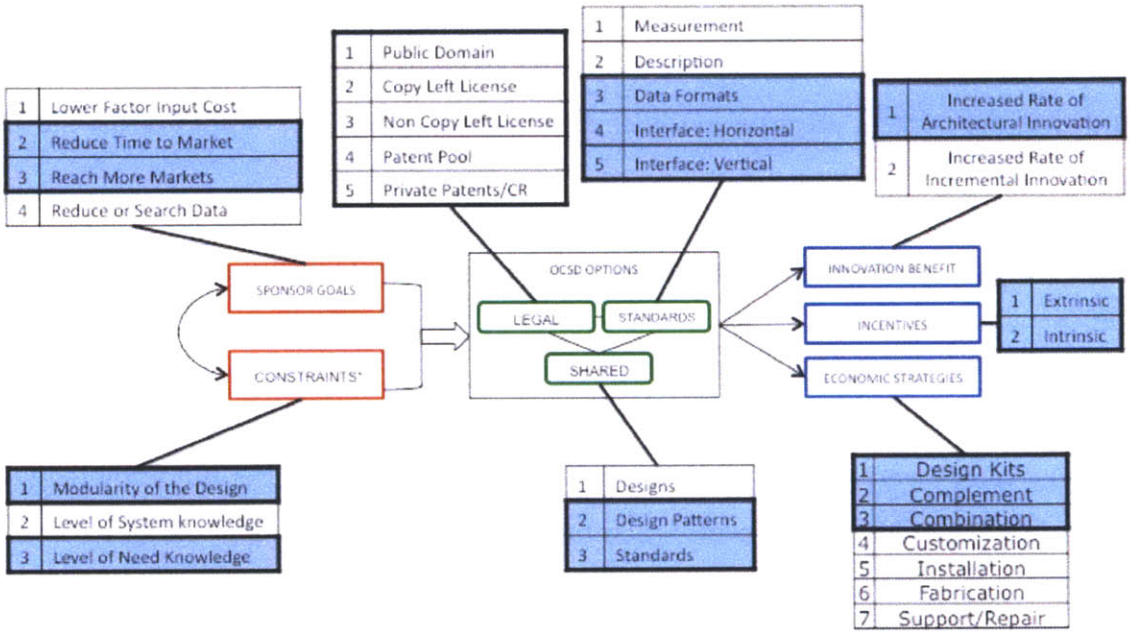


Figure 41: Architectural Development OCSD.

5.5.2 *Technical Background: Biobricks & iGEM*

The proposed OCSD strategy is best applied where the technology is well understood, and therefore vertical or horizontal decoupling of a technology is possible. While synthetic biology has not yet achieved reliable decoupling, it has created a conceptual framework with which to envision how such design might be possible in biotechnology. This should be reviewed in light of the proposed application of the strategy.

As described in Chapter 1 of this thesis, synthetic biology has created a conceptual framework with which to design genetic machines, based on modern engineering practice. Within the conceptualization, there are three distinct "abstraction hierarchies" for genetic systems: Parts, Devices, and Systems (Figure 42).

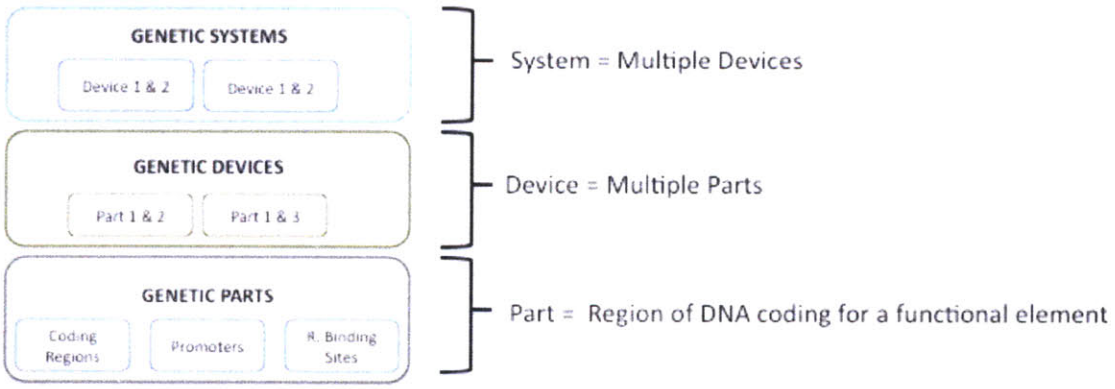


Figure 42: Abstraction hierarchy for synthetic biology.

Within this synthetic biology framework, genetic parts are units of DNA that serve functional purposes. They may be regions that code for proteins (coding regions), or regions that help with the basic transcription and translation of DNA (Promoters, Ribosome Binding Sites, etc). These parts can be "assembled" together to form devices that serve specific functions, and the devices in-turn can be used to create more complex systems. The assembly is accomplished via techniques of molecular biology such as cloning, polymerase chain reaction (PCR), etc.

Critical to this vision is the concept of idempotency (Knight 2005). That is, each assembly leaves the key structural elements of the components the same. An idempotent assembly (whether part-part or device-device) is itself a component which can be assembled with other parts or devices (Knight 2005). To accomplished idempotency, Synthetic Biologists have created an *assembly standard* called the Biobricks Assembly standard (Knight 2005). Within the context of the broader proposed OCSD framework, the biobricks assembly standard is a *horizontal interface standard* – it facilitates interfacing between elements at a given layer of abstraction.

If the vision of reliably re-assembled genetic parts, devices and systems is clear, it is not yet a reality. The extraordinary complexity of the living cell continues to belie biologists' attempts to easily manipulate genetic machines. The iGEM competition encourages teams from around the world to use the Biobricks standard to build

machines, and thereby slowly develop the knowledge needed to make the broader vision a reality. As discussed in Chapter 1, iGEM teams are encouraged to use the Registry of Standardized Biological Parts to use and submit parts. To date the teams have had mixed success.

5.5.3 Background on Proposed Sub-Domain for this Strategy: Biosensors

5.5.3.1 Biosensing

Before applying the Architectural Development strategy to biology it will be useful to specify the domain of application and to describe how iGEM teams have used the Registry of Standardized Biological parts to create designs within this domain. For the purpose of analysis let us fix the domain of application to sensors, understanding that there are diverse markets in which sensors might be used.

Sensing is an appropriate test case for the Architectural Development strategy in synthetic biology for a few reasons. First, a number of genetic modifications for biosensing have been developed already by iGEM competitors. Second, sensing is an information-based effort and, as noted in chapter three, this strategy is usually applied within the information-intensive industries. Finally, a range of different elements and compounds can be sensed, creating a variety of end-uses and markets.

By way of background, we note that the function of a sensor is to identify and measure an environmental stimulus. In its simplest form it is an input-output device that must identify a stimulus, translate it into a measureable output, and then communicate that output to another device (Figure 43).

Figure 43: Basic Functional Requirements for any sensor.

MFCs can be used as *part* of a biological sensor, via their ability to interface

biological phenomena with electronics. That is, the mechanisms which enable the cell to generate electricity via exo-cellular electron transport can be linked to mechanisms that sense environmental inputs. The question remains, how has this been accomplished in the past? And how should one conceptualize how to open such a system? The next section reviews iGEM sensing designs. The following sections use the options and constraints defined by the proposed OCSD framework and Architectural Development strategy to provide guidance.

5.5.3.2 Sensors made by iGEM Teams

Over the past seven years, the International Genetically Engineering Machines Competition (iGEM) has encouraged student teams to design and construct biological machines with the help of standardized parts shared through a registry. To date teams have had varied success completing their designs and adapting previous parts and systems to their designs. A number of teams chose to build sensors, given their simplicity, and one team recently proposed a Microbial Fuel Cell-based sensor. This section reviews some of the efforts of four such teams over four years.

Table 14: Four iGEM sensor projects from 2006 to 2009. MFC sensor assembly combines microbe-based sensors with exo-electric activity to create a fuel cell-based biosensor.

Project	Year	Team	Chassis
Arsenic Cell Sensor	2006	Edinburgh	E. coli
Lead Cell Sensor	2007	Brown	E. coli
MFC Sensor Assembly	2008	Harvard	E. coli; Shewennella oneidensis
Histamine Cell Sensor	2009	MIT	E. coli

Most of the four sensor teams here used a similar concept. They coupled the presence of an input stimulus with the "up regulation" of the *lac operon* in E. coli, which can be coupled to a change in the environment such as a pH change. The *lac operon* (Figure 44) is an assembly of DNA regions that regulate the metabolism of lactose. It consists of a *promoter* that determines when to express certain genes, three structural genes that help digest lactose, a terminator, and an operator. It has been used extensive in genetic engineering, and can therefore be easily incorporated in a signaling device that seeks to connect an input stimulus to some output.

Figure 44: Lac Operon (courtesy: Wikipedia.com)

The way in which the lac operon was used varies from project to project. Yet, it is possible to identify high-level contributions to the registry. Figure 49 provides a snapshot of the mean number of DNA parts (coding regions), Devices (Combination of Coding regions with ancillary promoters or RBS binding sites, etc) and Systems (combinations of devices).

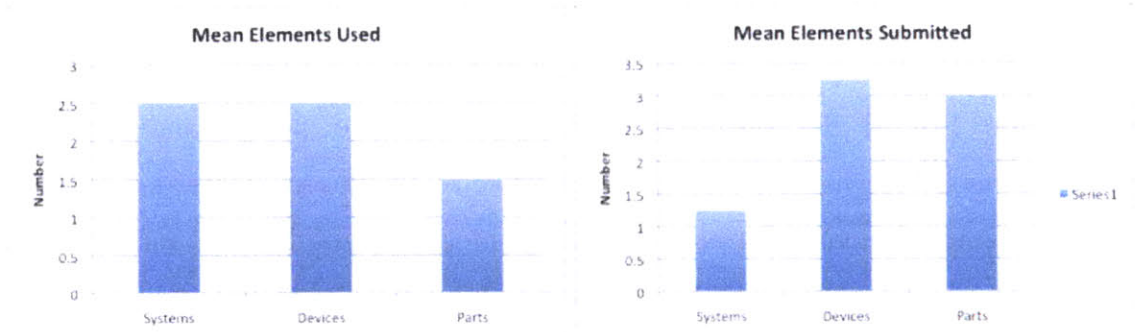


Figure 45: Mean parts, systems, and devices submitted and used among the four teams.

While the sample set is low, it does demonstrate that the number of parts used and submitted is fairly low, illustrating that the systems used to date are not very complex. The charts also indicate a slight trend towards using a greater number of systems than parts, but submitting a greater number of parts than systems. Finally, the figures indicate a similar range of parts, devices, and systems – 1 to 3 – whether they are used or submitted.

5.5.4 OCSD Inputs: Sponsor Goals, Constraints

The Architectural Development strategy (Table 11) was found to be employed most often by developers of a technology (rather than users or other beneficiaries). In this case, developers would be for-profit sensing teams, or foundations such as the Biobricks foundation. Let us assume for this description that the sponsor a non-profit organization, such as the Biobricks Foundations, with the goal of generally promoting innovation in synthetic biology. This is chosen because such sponsors are currently pursuing the strategy through mechanisms such as the Registry of Standardized biological parts.

With respect to constraints, the strategy is best applied when there is low knowledge of the range of potential end-uses of a technology but high knowledge of the underlying technology itself. In a real-world situation, then, one would select the strategy if these elements were present. We assume, based in part of the work of iGEM teams discussed above, that the basic elements of biosensors are fairly well understood.[18]

5.5.5 OCSD Design Variables: Standards, Sharing, Legal Mechanisms

The sponsors of this OCSD have a range of design options to chose from, as defined within the broader framework. These fall into three categories: (1) standards (2) what is shared (3) legal mechanisms.

5.5.5.1 Standards: Horizontal Standards, Portability and Vertical Decoupling

As illustrated in Figure 41, the Architectural Development strategy depends on three kinds of standards within the broader framework: (1) horizontal interface standards (2) vertical interface standards (3) data-formats (in the cases where data is exchanged between parts).

Horizontal interface standards occur between parts at a given layer of abstraction. The biobricks standard is an example of a physical, horizontal interface standard at the genetic level. Further horizontal interface standards might include functional (rather than physical) genetic interface standards. Standard interfaces between chassis cells (perhaps using a variation of quorum sensing communication systems) might also be develop if the range of chasses cells continues to increase.

The detailed VLSI case study presented in chapter 4 concluded that vertical decoupling between layers in a value chain or layers of abstraction is more important than horizontal decoupling for OCSD. This is because by decoupling

[18] It is not yet the case that the use of genetic parts, systems, devices in sensing, or any other domain, is well understood. We make this assumption in order to describe how the strategy would be applied, given better system knowledge.

layers, designers can re-use any elements in a lower layer while creating new designs to address new markets. For example, by creating application programming interfaces to web-services such as Google Maps, Google enables the creation of a wide variety of websites that incorporate the mapping service.

As noted in chapter 4, vertical decoupling occurs when designs at a higher layer of abstraction are portable across designs at a lower layer. For example, Mead and Conway vertically decoupled VLSI design representations from any particular fabrication facility by inventing scale-independent design rules that could be rapidly modified to fit any fab. Designs written using the Lamda Rules were portable across fabs.

What does portability mean in the context of biology? It simply means that designs at one layer can be ported across different devices. Thus, different parts can be used in many devices. More specifically, in this case it means that genetic designs can be ported across different exo-electric chasses, or different chasses can be ported across different microbial fuel cell sensors.

How can this be envisioned? Figure 46 maps the three sensor functions defined in Figure 43 to the biological context based on the iGEM sensor teams work. It illustrates that – conceived as a value chain or an abstraction hierarchy – the microbial fuel cell hardware serves the purpose of transmitting and communicating exo-cellular electron transport (cell signal) to a computer; different kinds of chasses cells with exo-cellular electron transport properties can be used in a given reactor; and different kinds of genetic systems that react to input stimuli might be designed for any given chassis cell.

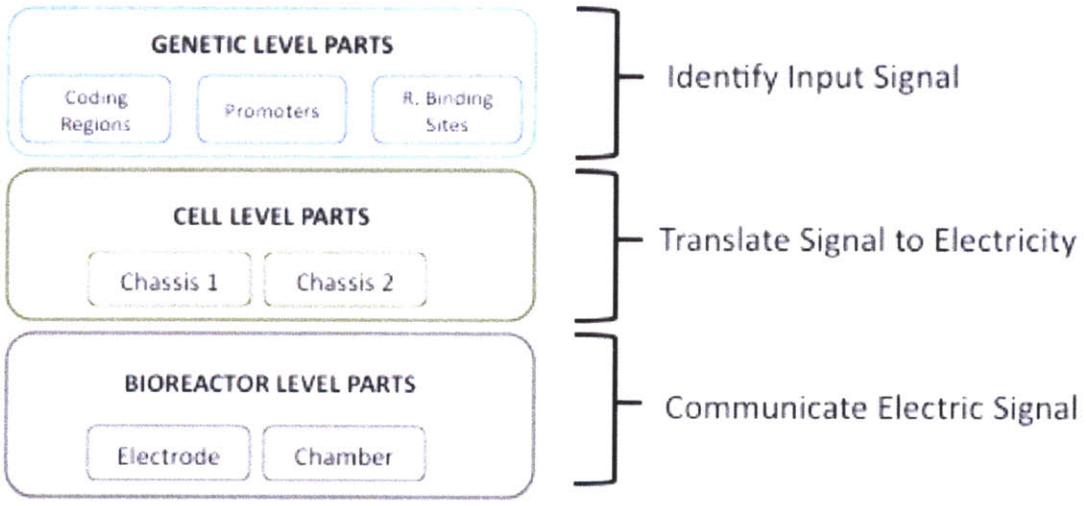


Figure 46: One conceptualization of the sensor stack for multiple applications. DNA Coding Regions (e.g. Genes), Promoters and Ribosome Binding sites are "designed" to sense different elements. Standard Chasses (e.g. microbial cell lines) operationalize the DNA parts and use exo-cellular processes to generate or interrupt current flow through a microbial fuel cell bioreactor.

To be clear, this is one of many potential "stacks" for biological sensors. But the representation clarifies how some concepts from other OCSD domains translate to the biological medium. For example "portability" in this context involves creating vertical standards that enable genetic-level designs to be implemented in a diverse range of chasses, or a chassis to be used in multiple MFC bioreactors.

The proposed biological sensor "stack" thus formalizes some of the questions raised above: Is it better to enable portability across hardware platforms, bacterial strains, or both? Within the context of Microbial Fuel Cells for sensing applications, this takes on specific meaning. For example, is it even feasible to use one common strain for all sensing applications, or do strains have a limit to the breadth of applications? If the former, can we turn commonly engineered chassis strains such as *E. coli* into exo-electric organisms, or is it more feasible to modify an exo-electric organism such as *G. sulferreducens* into a shared chassis cell? To what extent do the reactors need to be modified for each application and does this create the possibility for an open architecture approach in which the biology is public but the hardware is private?

As noted, to date the synthetic biology community has focused largely on the DNA-

Bacteria interface, or on abstraction hierarchies within the genetic level (parts, devices, systems).[19] However, as noted in (Moses 2002), a layered decomposition usually includes three layers, each of which might be further decomposed into three layers. Figure 47 clarifies this point by illustrating that the genetic abstraction hierarchy envisioned by synthetic biologists (Endy 2005) can be considered a decomposition within the broader hierarchy of reactors, cells, and DNA.

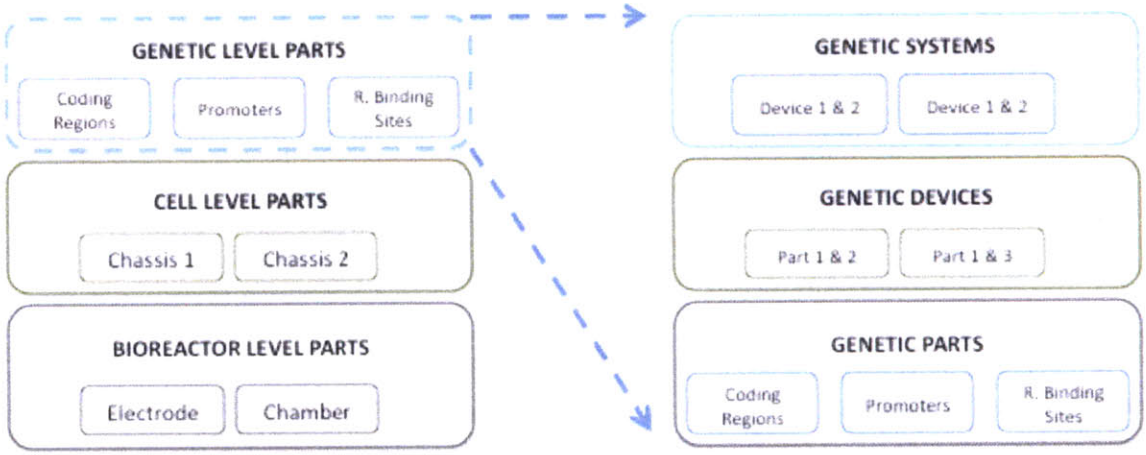


Figure 47: Three primary layers of abstraction in the sensor MFC value chain, versus the abstraction hierarchy created for synthetic biology. Genetic abstraction hierarchies represent a sub-partition within the genetic-level. What are the sub-partitions at the cell and reactor levels? Need there be any?

The figure suggests that, within the cell-level there may be multiple layers of abstraction that might be considered as well. This would be multi-cellular devices and multi-cellular systems. Knowledge of these elements in biology stems largely from our understanding of Physiology. The human body, for example, has multi-cellular devices called organs, and the body is a "multi-organ" system.

Whether a closer examination of physiology would benefit synthetic biology or not, it is clear that nature of vertical interface standards have not yet been fully explored. This points to a gap that should be filled were the Architectural Development strategy to be employed. Before discussing this, let us examine the remaining two

[19] While some have noted the importance of understanding the interactions between strain and environment Andrianantoandro, E., S. Basu, et al. (2006). "Synthetic biology: new engineering rules for an emerging discipline." Mol Syst Biol **2**., the standardization of this interaction has not be thoroughly explored. One exception is Kelly, J., A. Rubin, et al. (2009). "Measuring the activity of BioBrick promoters using an in vivo reference standard." Journal of Biological Engineering **3**(1): 4.

design variables proposed in the OCSD framework – sharing design patterns and the legal mechanism.

5.5.5.2 Sharing Sensor Design Patterns: Example from the iGEM competition

Vertical decoupling would enable more efficient sharing of DNA parts, devices, systems, as well as chasses and bioreactors. While it is not yet clear what level (or levels) might be opened, it is possible to examine the experience of iGEM teams that have re-used genetic parts, devices, and systems from the Registry of Standardized biological parts. The four teams analyzed here were described above, in section 5.5.3.2.

Further analysis of the four teams' projects demonstrates that some teams submitted many parts but no systems and vice versa. Figure 48 makes this point more clearly, by breaking down the number of parts, devices and systems submitted or used by each sensor team.

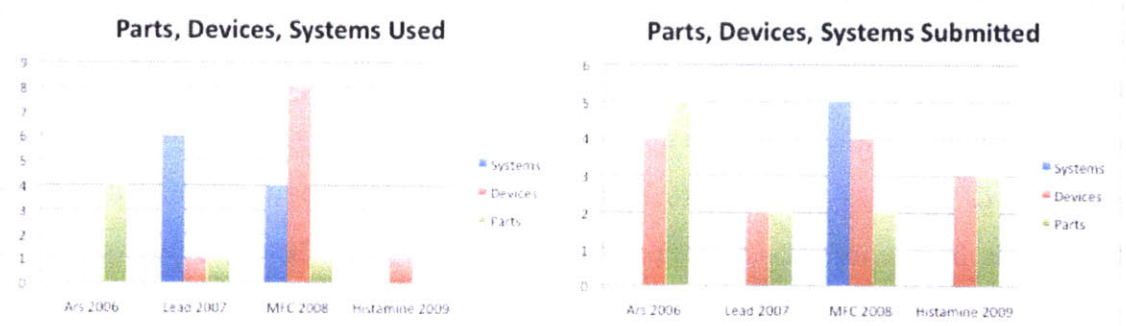


Figure 48: Parts, systems, devices used (left) and submitted (right).

The cursory data suggests a slight trend towards using more parts and devices from 2006 to 2008, while the number of parts and devices submitted remains roughly the same. This can be highlighted by aggregating the total numbers by year.

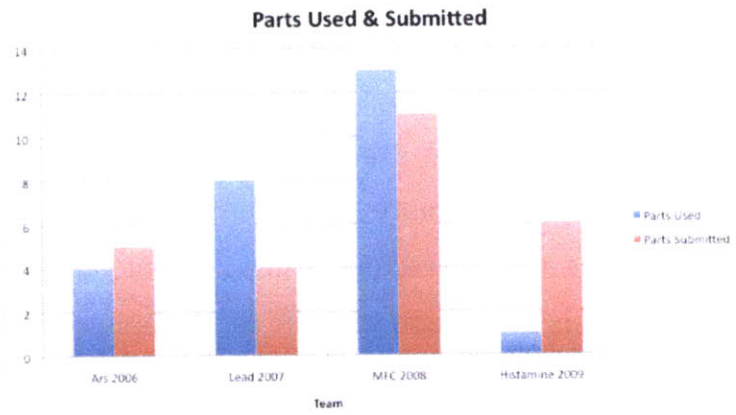


Figure 49: Total parts versus submitted by each team.

The data suggests that the parts and device swapping enabled by the registry is indeed useful for developing sensors. However, as indicated, below these numbers are not the actual results of the projects. In particular, few of the proposed sensors actually functioned as originally planned.

For example, the Harvard team attempted to use the mtrB gene, which codes for exo-cellular transfer genes in Shewenella, in E coli. However, it turned out to be toxic to *E. coli* (Bactricity 2008). They solved this problem by employing a two-cell system – Shewenella was used to create an output based on regulation of the mtrB gene, and *E. coli* was modified to create an output that Shewenella could sense. These basic problems persist throughout the iGEM teams' descriptions, indicating the difficulty in implementing even two-part systems in different chasses. From the perspective of portability, these experiences highlight the difficulty in decoupling DNA-level designs from chassis.

While it is difficult to draw definitive conclusions based on this analysis, we might tentatively assert that sharing chasses cells is currently more efficient that sharing DNA parts.

5.5.5.3 Legal Mechanisms

Similarly to the Incremental Development strategy, the proposed Architectural Development strategy can utilize any of the five potential legal mechanisms

described in Figure 41. However, the broader framework noted that sponsors of this strategy often keep an entire layer (or a few layers) open, while reserving proprietary development for lower or higher layers of abstraction. This suggests that at least for the sensor stack (Figure 46) a license might be created which keeps the hardware, cells or genetic systems open, together with the vertical interface standards (however they are ultimately defined).

Given the existence of The Registry, one might suggest that such a license should employ a copyleft-like clause forcing genetic parts compatible with the Biobricks standard to remain open, while enabling patenting of novel cell-lines and novel reactors. This, however, faces two problems. First, it is not clear what should be considered a novel chassis, versus a novel genetic system. Given that a novel Chassis includes novel genetic systems, we would at least need a clear description of "chassis" functions versus "application" functions.

A second concern with keeping the genetic level open is that, metaphorically, "genetic systems" are the equivalent of computer applications. In the computer OCSDs described in Chapter 3, *applications* are almost always proprietary, while lower level infrastructure – such as operating systems and hardware – is opened (either architecturally or completely). Opening the application layer creates an external incentive to contribute to the lower-level ecosystem. Even GNU/Linux, for example, now runs somewhat proprietary code on top of the GNU/Linux operating system as described in chapter 3.

Building on the computer industry metaphor, then, we rather envision a value-chain in which bioreactors could be assembled using an open architecture, while chassis were kept open via a copyleft type license. This would require open vertical standards between reactors and cell-lines. Again, this raises the question – what should be the functions that the open-cells and open-reactors carry out, in order to support the higher-level applications?

In the case of Microbial Fuel Cell biosensors, we have a partial answer. The bioreactors support cell growth, and provide electrodes to facilitate and sense exo-cellular electron transfer. However, at the cell level and DNA level, the answer was unclear. Should a chassis be a microbe like E coli – in which case one "application" includes the genetic systems for exo-cellular electron transfer like mtrB? Or should the chassis be Shewenella, with the applications being novel genetic systems for sensing environmental stimuli? Both solutions face problems with portability, as discussed above.

5.5.6 OCSD Outputs: Incentives, Innovation Benefit, Developer Strategies

Based on the discussion above, we can briefly summarize the "outputs" of the Architectural Development strategy within the proposed OCSD. First, the primary incentives will be both extrinsic and intrinsic. The intrinsic incentives, at least for this proposed application domain, include learning and fun within the context of the iGEM competition and other educational thrusts. External incentives include monetary gain and rewards associated with developing new products in combination.

The innovation benefit to the sponsors of the regimes (our assumption was that the sponsors included the Biobricks Foundation), include new parts that enrich the ecosystem, and new end-applications not previously envisioned.

Finally, we envisioned the sensor stack (Figure 46) as a three-tiered, layered hierarchy in which multiple chasses could be used across multiple MFC bioreactors, and genetic designs could be used across multiple chasses. Within this context, developer strategies will include selling "combinations" (Figure 41) of MFCs, chasses, and some genetic designs. Depending on what kind of license is used, different elements of these systems will be proprietary.

5.6 Conclusions: Using OCSD in Synthetic biology

This chapter applied the proposed OCSD framework, as well as two of the three

strategies – Incremental Development and Architectural Development – to the development of a novel technology called Microbial Fuel Cells. After briefly describing the technology, a particular sub-domain of application was chosen for each strategy. For Incremental Development, MFCs were applied to the specific problem of wastewater treatment. For the Architectural Development strategy, MFCs were applied to the multi-market domain of biosensing. Application of the strategies included two elements: First, the basic categories within the broader proposed OCSD framework (as constrained by each strategy) were described. Second, questions arising from this description were analyzed qualitatively.

The chapter therefore validated the utility of the proposed framework within the biotechnology domain. In doing so, it also enabled some general conclusions about OCSDs in biotechnology. The application also raised some important questions for each strategy. These are discussed below.

5.6.1 Incremental Development Aided by the Natural Modularity of Biology

Application of the Incremental Development strategy identified important co-dependencies between the bio-film and the reactor that may need to be overcome. It also suggested that, given the nature of the biological medium, we might expect a dynamic in which consultants and engineers sell construction and maintenance services to water treatment customers, and then biological engineers sell proprietary microbial designs to this fixed infrastructure. This dynamic is greatly facilitated by the natural modularity of biology. In this case, because cells are physically distinct from the reactor, we can easily envision upgrades to an installed system via novel cell lines. For initial developers or users to benefit from this dynamic, a license should be constructed that requires sharing of performance data across reactors, enabling users to continuously improve their operations.

Application of the Incremental Development strategy also raised important questions that can now be answered in more detail in the following chapter. In particular, if an entire architecture were distributed, given the potential decoupling

between reactor and biofilm, how would a sponsor determine whether to give away microbes versus reactors? What would be the impact on innovation?

5.6.2 Architectural Development via Portability & Appropriate Layering

The Architectural Development strategy was described and analyzed with the help of data from the iGEM competition. The iGEM teams' experience highlights both the promise and the challenges associated with library-based design in the biological context. Each team used, on average, between one and three parts, devices and systems from the registry. This suggests that transaction costs were indeed reduced by creating a library of parts. However, most designs did not function as expected. This illustrates the current limitations of rapid prototyping of biological designs.

Difficulties sharing parts among chassis cells highlights problems with portability in the medium. It seems that the field might benefit from consideration of the range of applications that each chassis might support, as well as the level of abstraction most appropriate for sharing in the biological medium. For example, the MFC teams found that resorting to higher levels of abstraction – in which whole chassis cells are "parts" – enhanced development. By changing the level of abstraction the requirements for portability will also shift – in this case from exchanging DNA between cells to exchanging cells between *reactors*. As with a number of examples described previously, however, increased portability created by raising the level of abstraction often results in a decrease of performance. In this case, decreased performance manifested itself through slower sensor response.

Application of the Architectural Development strategy also highlighted that the prevalent abstraction hierarchy envisioned by synthetic biologists – genetic parts, systems, devices – might be broadened to include reactors, cell-lines, genetics (Figure 47). It would appear, for example, that the cell-reactor interface and the DNA-cell interface is more important from an OCSD perspective than the partitioning of DNA designs into parts, devices, systems. This notion builds on the analysis of VLSI design presented in chapter 4, which found that *vertical decoupling*

could be accomplished in the absence of library-based design.

Further, the three layers of abstraction at the genetic level might be matched by three layers of abstraction at the cellular and reactor levels. For example, cells can be formulated into organs and the organisms. Our knowledge of physiology could help elucidate the relationship of form and function at these levels. One might argue that solving design challenges at lower levels of abstraction is first needed. However, as noted, some of the iGEM sensor teams found that designing at the cell-cell level was easier than at the genetic level because it eliminated the need for portability of DNA designs across different cell lines.

Finally, application of the Architectural Development strategy suggested that the current emphasis on keeping the genetic layer open might be revisited. Many OCSD examples that follow the Architectural Development strategy kept the "application layer" closed. In many of these cases one or more layers were opened below the application layer. For example, GNU/Linux and MySQL are open layers of abstraction 'below' the end-application layer. Web services, and other proprietary applications, can be developed on these layers using APIs. This creates powerful extrinsic incentives for developers of complimentary technologies to develop and maintain the open resources below.

Figure 50 illustrates schematically the opening of only the chassis layer in a microbial fuel cell OCSD. In this case different proprietary MFC bioreactors interact with one open chassis population that "runs" different DNA programs.

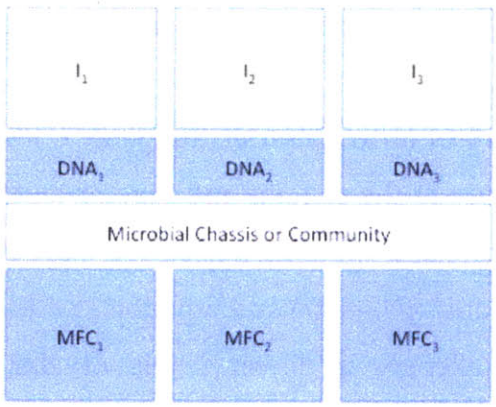


Figure 50: The abstraction of the microbe layer from the MFC layer. MFCs and Inputs vary only within specified bounds in this strategy. Therefore the microbe layer has fixed functional requirements.

More generally, it should be noted that the vertical and horizontal decomposition of complex systems is itself a design problem, at the discretion of engineers. Currently, synthetic biologists have great freedom in defining this abstraction hierarchy. The broader framework and strategies suggests, at least, that an approached which recognizes the distinction between layering and horizontal decoupling may be useful.

5.6.3 Energy versus Information

Finally, it is worth noting that the case studies from chapter three could be mostly, though not fully, divided into energetic versus information-intensive. Where does biology fall? This chapter suggests that it can be either or both. Microbial Fuel Cells treating wastewater are an energetic technology. The OCSD sponsor goal in this case involved improving performance along one, or at most, two dimensions. However, biosensors clearly have the goal of transmitting information. This fact clarifies that the energy-information dichotomy described previously is not binary. Rather, it is a spectrum along which different technologies fall.

The broader OCSD framework suggests that interface standards between parts and between layers of abstraction usually hinder performance. This might suggest that for some applications in which throughput or rates are important – fermentation to biofuels, for example – only the *Incremental Development* strategy would be helpful.

However, biotechnology has a distinct different with traditional technologies in its ability to evolve. Within this context, one can envision a scenario in which vertical interface standards are used speed up development, and then modified to enhance performance during operation.

Regardless of which strategy, or blend of strategies is pursued, this chapter demonstrates that the broader framework should help guide decisions regarding standardization and non-technical aspects of system development.

6 Quantitative Methods for Exploring Open Design

6.1 Introduction

The previous chapter outlined how the strategic framework can be applied to one particular biotechnology, including qualitative benefits and limitations associated with two strategies. This chapter proposes a quantitative method and an analytical framework with which to evaluate the impact of opening microbial fuel cells. The method is based on multidisciplinary design optimization and pareto analysis. As described in more detail below, it has four steps: 1. Create a multidiscipline model 2. Identify feasible bounds on relevant parameters and create a Pareto plot 3. Identify and visualize sub-sets within the design set that correspond to standardized design variables 4. Calculate losses in benefit/cost associated with these subsets against the pareto-optimum solutions.

The method can help visualize and quantify the implications of standardizing and thus opening certain elements of the design. Interpretation of these results can guide standard setting, and can help determine design variables that are better left closed based on the objectives of the sponsor.

Because the design variables are bound using physical constraints – rather than knowledge of internal system function or internal constraints – the method can be employed without significant knowledge of potential design solutions, and continually refined as new system knowledge is generated. It can therefore be employed within the first strategy outlined in the broader strategic framework, in

which a sponsor with low system knowledge seeks to develop a technology with a specific and well defined market needs.

The chapter is organized as follows: First, a background is provided for quantitative modeling of microbial fuel cells, including high level objectives and constraints. Then the fuel cell model is developed and defined. This is followed by application of the proposed method: (1) Create a multidiscipline model (2) Identify physical bounds on design variables and create a paerto plot (3) Partition the design set based on standardization and (4) Visualize and quantify performance impact. Overall, the chapter demonstrates that it is possible to quantitatively evaluate the impact of standardizing elements of a biological product architecture.

6.2 Microbial Fuel Cell Design Objectives

The proposed method will be applied to the design of microbial fuel cells treating water and generating electricity. Background on this technology was provided in the previous chapter. More formally, we define the objective of MFC design for wastewater treatment as the simultaneous maximization of the energy density, $\boldsymbol{P_D}$ (Watts/m^3) and Biological Oxygen Demand (BOD) removal rate, $\boldsymbol{BOD_r}$ (KG/m^{3}Day) at minimum cost, $\boldsymbol{C}$ (\$/m^3). This constitutes a multi-objective optimization problem which could be solved using scalar or pareto methods (De Weck 2004). For the model presented here, we combine performance and cost into a cost/benefit for each performance parameter. Dividing P_D by C and gives us the Watts per dollar, $\boldsymbol{P_{CV}}$ (W/\$). Dividing BOD_r by C yields the KG BOD removed per day per dollar of capital investment, $\boldsymbol{BOD_{CV}}$ (KG/Day\$). The MFC design objective then becomes a pareto-optimization problem with the following objectives:

$$\text{Max } [P_{cv}, BOD_{cv}]$$

Where:

$$P_{CV} = \frac{P_D}{C}$$

$$BOD_{CV} = \frac{BOD_r}{C}$$

Given a set of design variables and subject to a number of constraints and parameters based on the nature of the reactor, electrochemical processes, biological processes, input and operations.

6.3 Constraints and Losses in Typical Fuel Cell Modeling

MFC optimization must take into account factors associated with traditional fuel cells, as well as specific constraints associated with the biological catalysts. The most important curve associated with a fuel cell is the ***Polarization Curve*** that relates the voltage of the cell to the current being drawn from the cell. Figure 51 below illustrates such a curve. The drop in voltage as the current increases is due to losses in the cell.

The power extracted by any circuit is dictated by Ohm's law, P = IV where P is the power in Watts, I is the current in Amps and V is the voltage across the electrodes. The cell voltage is dictated by the difference in free energy of formation (Gibbs free energy) of the anodic and cathodic reactants, minus losses that occur prior to and during current creation. For a hydrogen fuel cell, the maximum theoretical open circuit voltage (OCV) is 1.482 volts corresponding to the differences in the enthalpies of H_2 and H_20. When standard operating pressures and temperatures are accounted for, this drops to 1.23 Volts, yielding a maximum theoretical efficiency for a Hydrogen-Oxygen proton exchange membrane fuel cell of 83% (Barbir 2005). In this case efficiency is defined simply as the operating voltage divided by the max OCV.

In reality, the operating voltage of a fuel cell will be much less than its theoretical voltage, due to losses that occur with current. The challenge in optimizing a fuel cell

from a strictly power density perspective involves finding designs that minimize these losses.[20]

Some of the most important constraints on the optimization problem include the losses in the cell. Electrochemists have identified three primary kinds of losses. ***Activation losses*** result from energy lost as heat for initiation of the oxidation reaction – they dominate at low current densities, and are important throughout the operating current densities. ***Ohmic Losses*** result from resistance of ion conductance through the membrane and electron conductance through the electrodes, as well as other contact resistances. They can result from solution chemistry of the electrolyte and various contact losses. Ohmic losses dominate in the regions where maximum power is generated, and are thus crucial to overall design. ***Concentration (or mass transfer) Losses*** result when the flux of reactants to the electrodes limits the reaction rate. These various losses conspire to reduce the operating voltage of the cell – to about 0.7 Volts for a hydrogen-oxygen cell (Barbir 2005), and lower in a MFC.

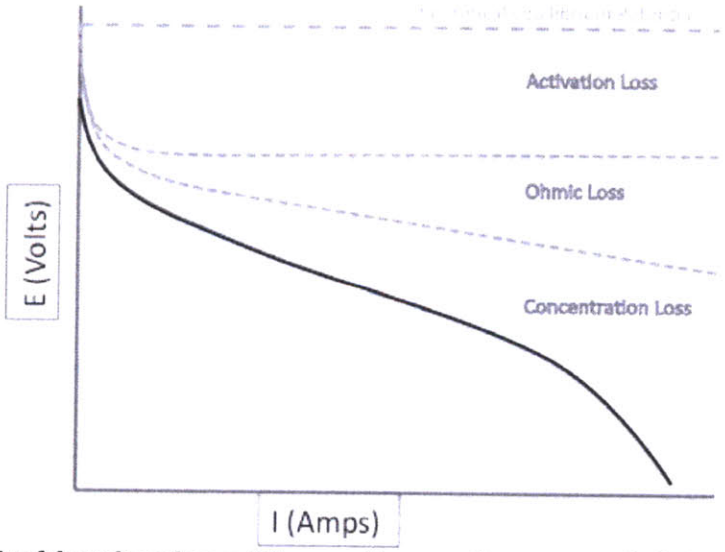


Figure 51: Schematic of the voltage losses (E) versus the operating current of a fuel cell. The three losses sum together to create the total Over-Voltage. (source: created on PowerPoint by the author)

[20] Other factors are also important for performance, such as columbic efficiency. These are addressed below.

The nature of these losses is now discussed in more detail before describing the model.

6.3.1 Ohmic Losses

Ohmic losses are caused by various sources of internal and external resistance. For analytical purposes, the internal resistance can be broken into constituent elements as described in (Fan, Sharbrough et al. 2008):

$$R_{int} = R_a + R_c + R_m + R_e$$

Where:

$R_a \equiv$ Anode Resistance due to bioelectrochemical reactions at anode
$R_c \equiv$ Cathode Resistance due to limitations of chemical reactions at cathode
$R_m \equiv$ Resistance of the PEM membrane
$R_e \equiv$ Resistance of the electrolyte

We can visualize these components of resistance in Figure 52.

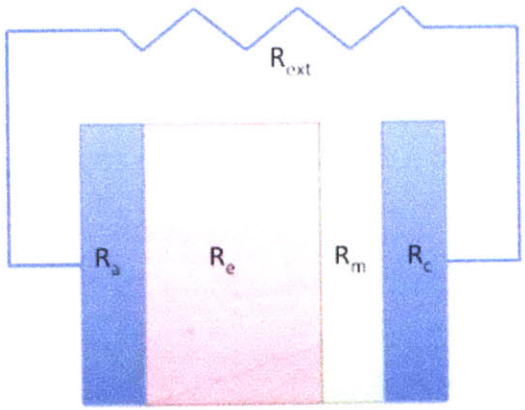


Figure 52: Schematic elements of internal resistance.

It is difficult to define these components of resistance independently of a specific microbial fuel cell, or without empirical observation. For example, R_a will be due to a host of factors including the interaction of the bacteria with the anode. However, we can define some of the values as a function of some properties of the MFC architecture, such as reactor spacing, L, and further specify the variables as the design develops.

By definition, the resistance of each element in the "stack" is defined as:

$$R = \frac{l \cdot p}{A}$$

Where l is the length, A is the cross-sectional area, and p is the resistivity of the material measured in Ωm. The resistivity of each component will be a function of the material used and the geometry. Also, we note that MFCs have an interesting property in that the electrolyte is also the fuel in most cases. If the fuel does not flow in a given direction, then the two driving forces affecting resistance will be the *diffusion* of ions in the liquid and the conductivity of the *solution*.

Once we have calculated our internal resistance we can transform it into a voltage loss as a function of current using Ohm's law.

$$V_{ohm} = I \times R_{\text{int}}$$

Where V_{ohm} refers to the drop in voltage due to ohmic losses.

6.3.2 Activation Loss

Activation polarization is caused by limitations in the natural rate of the reaction at an anode or a cathode. It is dependant on the *exchange current density*, defined as the current at the electrode at equilibrium in A/cm^2. The activation loss at each electrode can be calculated by the following equation presented in (Barbir 2005):

$$V_{act} = \frac{RT}{\alpha F} \ln(\frac{i}{i_o})$$

Where:

V_{act} = The loss of voltage due to activation polarization
R = gas constant (8.314 J/mol K)
T = cell operating temperature (K) – Room temp is 298K
α = transfer coefficient
F = Faraday's constant (9.64853e4 coulombs / mole of electrons)
i = The operating current
i_o = The exchange current

The transfer coefficient is a complicated term in and of itself, corresponding to the speed of the rate limiting step in the chemical reaction (Barbir 2005). It is usually about 0.5, however this can vary. Simplifying this equation by assuming normal temperature and pressure the equation becomes:

$$V_{act} = \frac{0.0236}{\alpha} \bullet \ln(\frac{i}{i_o})$$

We still need to know the exchange current in order to quantify this number. Currently, the exchange current at room temperature for a hydrogen reaction for a platinum electrode can vary between is approximately 10^{-4} and 10^{-9} A/cm^2 Pt (Barbir 2005). Depending on the catalyst specific area and the loading concentration, it can therefore rise to about 10^{-2} A/cm^2 of the electrode surface area. The exchange current for the biofilm anode is very hard to know without measuring. If each bacterium in a biofilm is considered a site of reaction, then it will be a function of the density of bacteria – or the specific surface area of the microbes – as well as the respiration rate of the bacteria. Increasing the exchange current could therefore be an important objective of biofilm design. This is addressed in the actual model below.

6.3.3 Concentration Loss

Concentration losses are to limitations associated with reactant diffusion to and from the surface of electrodes. If reactants are consumed quickly, a concentration gradient forms between the "bulk" fuel and the point of consumption. Figure 53 illustrates the concentration of anions create by the catalyzing biofilm. It shows that because anions are consumed at the cathode there will be a gradient towards the cathode. There will be a similar gradient, in reverse, with respect to the fuel moving towards the anode.

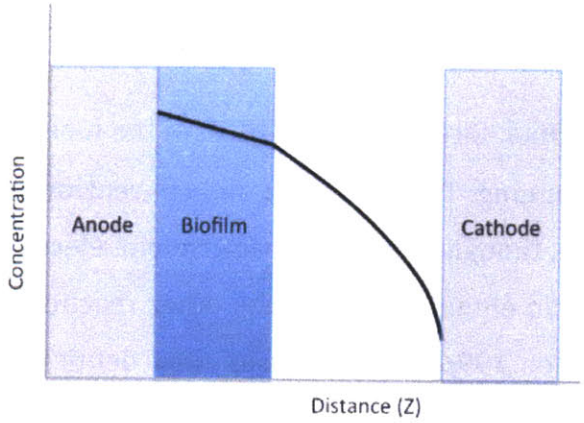


Figure 53: Concentration gradient in the fuel cell.

If reactants are consumed faster than they can diffuse, the concentration of reactants at the surface drops to zero. Therefore there is a ***limiting current density***, caused by concentration and diffusivity factors, at both the anode and the cathode. Concentration losses are thus largely a function of the limiting current density (Barbir 2005). Babir notes that this can be represented as follows(Barbir and Gomez 1997):

$$V_{conc} = \frac{RT}{nF} \ln\left(\frac{i_L}{i_L - i}\right)$$

Where i_L is the limiting current density (mA/cm2) of the anode or cathode. In the equation, n, represents the number of moles electrons liberated per mole substrate consumed (Barbir 2005). There are various ways to calculate the limiting current density itself, which can be explored. For our modeling purposes, we note that the limiting current is not the driving loss in microbial fuel cells, and we can therefore assume a relatively high number without affecting results significantly (Logan 2008).

6.3.4 Combining Losses to Create a Polarization Curve

The losses in the fuel cell determine how the voltage of the cell will drop as current is drawn during operation. The voltage loss will be a linear sum of the different losses (Barbir 2005).

$$V_{cell} = V_o - V_{ohm} - V_{act,a} - V_{act,c} - V_{canc,a} - V_{conc,c}$$

In this case V_o is the Open Circuit Voltage (OCV) of the fuel cell, which is the voltage when no current is running. The activation and concentration losses are separated by anode and cathode, though in practice because one electrode usually defines the limiting current, we can eliminate one or the other reaction chamber for each loss (Kordesch and Simader 1996). These losses now define a polarization curve as described above. Using the curve we can extract critical values such as ***Power***, ***Efficiency***, and ***Rate of BOD*** removal and thus we can frame the optimization problem.

It is important to note that, various other parameters will contribute to these losses. As described in (Logan and Regan 2006) factors such as microbe type, electrode spacing, geometry, materials, will determine the nature of these losses. These are now articulated, as they relate to the various losses described above.

6.4 Formulating the Pareto-Optimization Model for Microbial Fuel Cells

The previous discussion described the critical elements that define a fuel cell polarization curve. The three major classes of loss are equally applicable to Chemical and Microbial Fuel Cells. However, additional factors must be included in order to adequately capture the differences between biological and chemical kinetics. In MFCs, the open circuit voltage, activation potential, limiting current, transfer functions, and other parameters discussed above are all a function of properties of the biofilm populating the electrodes and its interaction with the reactor. This section describes a multidisciplinary model created to account for these biological properties and their interaction with the chemical constraints identified above.

6.4.1 Model Architecture and Assumptions

Recent work that has sought to extend modeling of chemical fuel cells to the microbial regime can be used as a basis for this model (Kato, Torres et al. 2007;

Picioreanu, Head et al. 2007; Torres, Marcus et al. 2008). As a general matter, we can note that, somewhat conveniently, specific parameters identified with each kind of loss above map in more or less straightforward ways to biological properties. The concentration and activation losses will be due primarily to the biological catalysts and the ohmic losses will be due primarily to the reactor architecture through the internal resistance.

For the purpose of creating his model we assume a very generic microbial fuel cell that consists solely of a bio-anode, a membrane, and a platinum air-cathode separated by a distance, L. This simple model is illustrated in Figure 54 below. Organic matter is oxidized at the biofilm anode, generating ions which travel through the fuel itself to the air-cathode. In reality, of course, there are infinite geometries that the cell might take, but these are less important to the method described below and the results of the model.

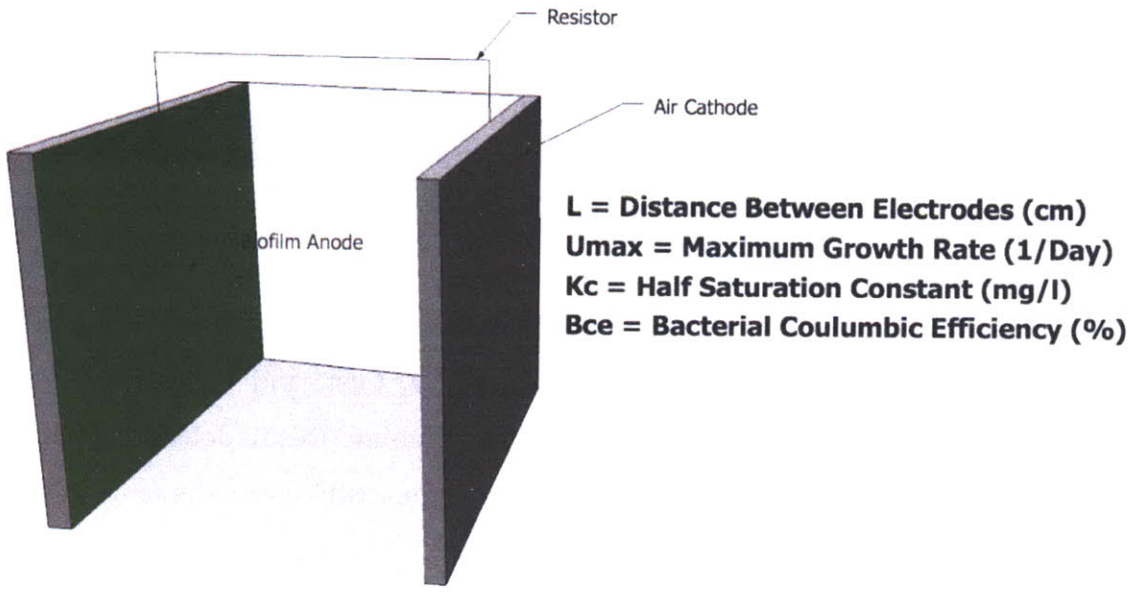


Figure 54: Simple flat-electrode model of a Microbial Fuel Cell used for this chapter. Design variables are listed on the right. U_{max}, K_c, and B_{ce}, will all be properties of the biofilm affecting cell performance through the Exchange Current, as a function of the input fuel.

The model therefore includes three basic elements, as described in chapter 6. The biofilm portion of the model will primarily affect the exchange current and limiting current, and therefore the activation and concentration losses. We assume that the length of the reactors and resistivity of the electrolyte are the primary drivers of Ohmic losses. Finally, the cost, as described below, is a combination of fixed CapEx and a variable amount which depends on the total area of the electrodes per unit volume. Internal resistance and cost can therefore be defined within the reactor model.

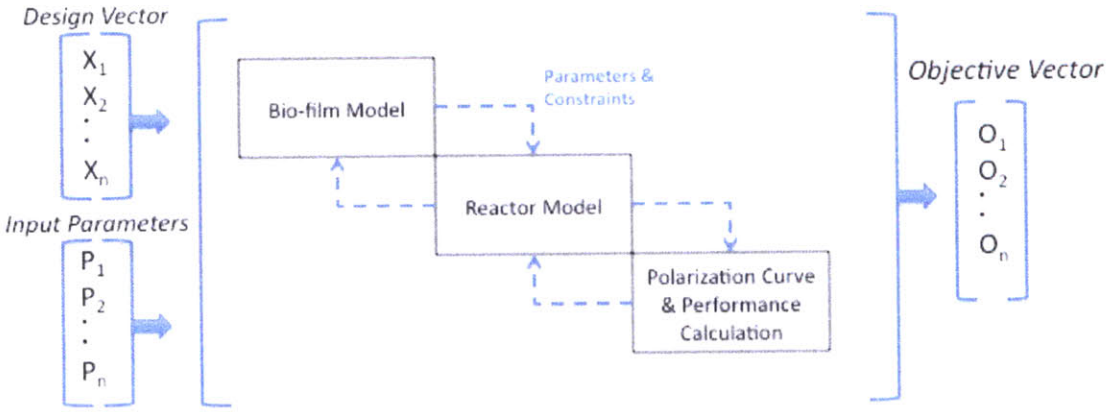


Figure 55: Schematic of the generic multi-objective design problem for MFCs.

Figure 55 provides a schematic for the basic elements of this model. Design variables, as described below, are varied as inputs. Input parameters including fuel concentration, materials costs, external resistance, and other factors are considered input parameters which might one day become design variables. These feed into three main modeling elements. The Biofilm model defines exchange and limiting current densities, the Reactor model defines OCV, Cost, and Ohmic losses; the Polarization Curve and Performance Calculation combines this to determine benefit-cost values for each design. The next few sections describe these elements in more detail as they apply.

6.4.2 Open Circuit Voltage in the MFC Model

While Hydrogen Fuel cells have a theoretical OCV, $\mathbf{V_T}$, of about 1.23 Volts at standard operating temperature and pressures, the actual OCV is substantially lower. For Microbial Fuel Cells with bio-anodes and platinum cathodes we can assume an OCV

based on oxidation of Acetate instead of Hydrogen (Logan 2008). The couple in that case has a maximum theoretical OCV of 1.105 V (Logan 2008). There will be several losses to this OCV before current is drawn however. The operating voltage of the bacteria will be slightly above that of the Acetate oxidation and will therefore cause another OCV loss of $\mathbf{V_B}$. Further, there will be crossover current, resulting a further OCV loss of $\mathbf{V_I}$. These OCV losses will not have a large impact on the nature of the results, other than to lower both BOD reduction rate and Power. We therefore model them as a simple loss of 0.2 volts each. The open circuit voltage used in the model is then:

$$V_n = V_T - V_b - V_I$$
$$V_n = 1.105 - .2 - .2 = 0.705$$

6.4.3 Modeling Ohmic Resistance in the Simplified Reactor

For modeling purposes we assume that Ohmic resistance is a function of the resistance of the electrolyte plus the external resistance. Electrolyte resistance, is a function of the resistivity of the electrolyte, in Ohm-cm, as well as the cross sectional area of the anode.

$$R = \frac{(L \times p_e + R_{ext})}{A_A}$$

Where $\boldsymbol{R}$ is resistance faced by a square centimeter of Anode area, $\boldsymbol{L}$ is the distance between anode and cathode, p_e, is the resistivity of the electrolyte/water in Ohm-CM, R_{ext} is the external resistance and A_A is the area if the anode in square centimeters. One might note that if R_{ext} is created by an external wire, its resistance should not decrease as a function of the anode surface area. The reason it is placed in the numerator here is that the current we will use is a current density of mA/cm^2 electrode area. Therefore, we need to find the resistance faced by a given cm of electrode area. Further the resistivity of wastewater does not drop indefinitely with increased surface area, since it is due in part to the "pronation" and migration of ions (Torres, Marcus et al. 2008). Therefore, taking these two factors into account, we made the simplifying assumption that the external resistance is

evenly distributed across the electrode. These assumptions should probably be verified experimentally, or double-checked, but they yield the correct relationships needed for the present modeling purposes.

Finally, because we are here concerned with the *volumetric* power the anode area will itself vary with L to maintain a cubic meter volume. If L is in centimeters, the Anode area in centimeters is:

$$A_A = \frac{10^6}{L}$$

Combining equation yields:

$$R = L \times (L \times p_e + R_{ext}) \times 10^{-6}$$

When running the model, R_{ext} was generally kept at 500 Ohms and p_e was generally kept at 1500 Ohm-cm, the approximate resistivity of polluted water (Hammer and Hammer 2008).

6.4.4 Volumetric Cost

Cell cost in the model is estimated as a function of the total electrode area. The most expensive electrode elements will be the platinum cathode and the membrane which, in our model, will be the same size as the anode. We further assume that there is a fixed cost for electrical equipment and software which is independent of the reactor size. In this case, total capital expenditure will be:

$$C = A_A \times C_m + C_{Anc}$$

Where A_A is the area of the electrode, C_m is the cost per square meter of membrane, and C_{Anc} is the fixed CapEx of ancillary equipment. Of course, in reality the membrane may or may not be the same size as the electrode to which the reactor is sized, but our simplified model the anode, cathode, and membranes are all the same

size. Because we are concerned with volumetric cost density, the area of the electrode will itself be a function of the spacing between electrodes as described in the previous section.

Finally, realistically, the cost per unit area of the electrode will tend to fall as the electrode area per unit volume rises, due to economies of scale – assuming a fixed volume. Economies of scale suggest that materials and manufacturing costs should fall as a power law:

$$C_m = a \times A_A^{\,W}$$

Where ***a*** and ***W*** are parameters dictated by the manufacturing process and material properties – "**a**" can be thought of as the cost one meter squared of membrane. Combining these three equations, then, we can create a simplified cost model:

$$C = (\frac{100}{L}) \times (a\frac{100}{L})^{W} + C_{Anc}$$

This reduces to:

$$C = a(\frac{100}{L})^{W+1} + C_{Anc}$$

We can see that the cost is primarily driven by the membrane and the distance between electrodes. In most of the runs of the model values of 100, 0.5, and 1000 were used for ***a***, ***P***, and C_{Anc} respectively, while L varied as a design variable.

6.4.5 Exchange Current and the Electric Biofilm

As the anode catalyst the biofilm will have a tangential influence on the internal resistance of the reactor, but a material impact on activation and concentrations losses. Activation losses are largely a function of the exchange current density, I_e, while concentration losses are largely a function of the limiting current density, I_{lim} (Barbir 2005). The Limiting current density is a function of the rate of diffusion of food ("fuel") into the biofilm. Simple calculations suggest that for the Microbial Fuel Cells developed to date, operating regimes have been far from the limiting rate at

which diffusion could theoretically occur (Logan 2008). Variations in limiting current caused by biofilm structure and fuel concentration are therefore ignored for the present modeling purposes, and a high limiting current of 10 mA/cm^2 is used throughout the calculations. This could be revisited in further research, however.

The exchange current density is defined as the rate at which a reaction proceeds at equilibrium (Barbir 2005). We propose that this is analogous to the natural rate at which bacteria in a biofilm consume substrate and respirate. We can therefore define the exchange current density using bacterial kinetics. In particular, the rate at which bacteria consume substrate is related to the rate at which a population grows, which can be defined using the monod equation (Logan 2008):

$$u = \frac{U_{\max} \times Conc}{K_c + Conc}$$

Where:

u = Specific growth rate in unit cell mass per day
U_{max}= Maximum growth rate in unit cell mass per day day^{-1}
Conc = Substrate Concentration in mg/L
K_c = Half Saturation constant in mg/L

The monod equation defines a relationship between input concentration and growth rate for a colony of bacteria. Given an initial amount of bacteria per square meter, Mib, the growth rate will be u times Mib per unit time. The mass of the bacteria per unit can be estimated based on the thickness of the biofilm, Tb, and the dry weight of the bacteria X. If the thickness of the biofilm is given in centimeters, the initial volume in liters per square cm of the electrode will be 0.001* Tb. Thus, given a dry weight per liter of X, and a biofilm thickness Tb the mass growth rate, Grate, of the population in mg per day per cm squared will be:

$$G_{rate} = u \times 0.001 \times T_b \times X$$

The amount of substrate consumed per unit area of the electrode can be inferred based on the yield of bacteria per unit substrate consumed (Logan 2008). Different bacteria have different yield rates, but in general the yield will be the inverse of the Columbic Efficiency of the bacteria, B_{ce} (Logan 2008). That is, we can make the simplified assumption that the electrons in the substrate are either used for growth or for energy – in the former they increase bacterial mass, in the latter they are consumed by the electrode or contribute to losses in the cell.

We can use the growth rate of the bacteria, together with the Columbic Efficiency of the bacteria, B_{ce}, and the atomic weight of the fuel to estimate the number of moles of electrons liberate through oxidation and transferred to the electrode. If we model the pollutant in the water as Acetate, the molecular weight is 136, and 8 moles of electrons are generated for every mole of acetate consumed. In that case, the number of electrons converted to current each day, $E_{current}$, is:

$$E_{current} = \frac{\frac{G_{rate}}{Yield} \times B_{ce} \times 8}{136}$$

In this case the Yield is the gram bacteria produced for every gram substrate consumed. In this equation, given the units used so far, one should divide the growth rate by 1000 to convert it to grams rather than mg, since atomic weight is a function of grams. Finally, we can now convert this to a current in Amps/cm^2 using Faraday's constant and converting days to seconds:

$$I_{ext} = \frac{E_{current} \times F}{60 \times 60 \times 24}$$

Putting all of the equations together we can reduce the previous equations to the following (including the division by 1000 to convert mg to grams mentioned above):

$$I_{ext} = 6.8 \cdot 10^{-13} \frac{U_{\max} \times Conc \times T_b \times X \times Yield \times B_{ce} \times F}{(K_c + Conc)}$$

While exact numbers for some of these parameters must be developed experimentally, we can make some assumptions to bound the likely exchange current, as described below.

6.5 Simulating the Polarization Curve and Quantifying Benefits

The various losses described here can be used to simulate a polarization curve which defines how the voltage in the cell drops as current is drawn. This curve can then be used to calculate the benefit of the cell in terms of Power output and BOD reduction rate. This section presents a simulated polarization curve using the model described above and realistic numbers. It then describes how Power and BOD reduction rate are derived from this curve.

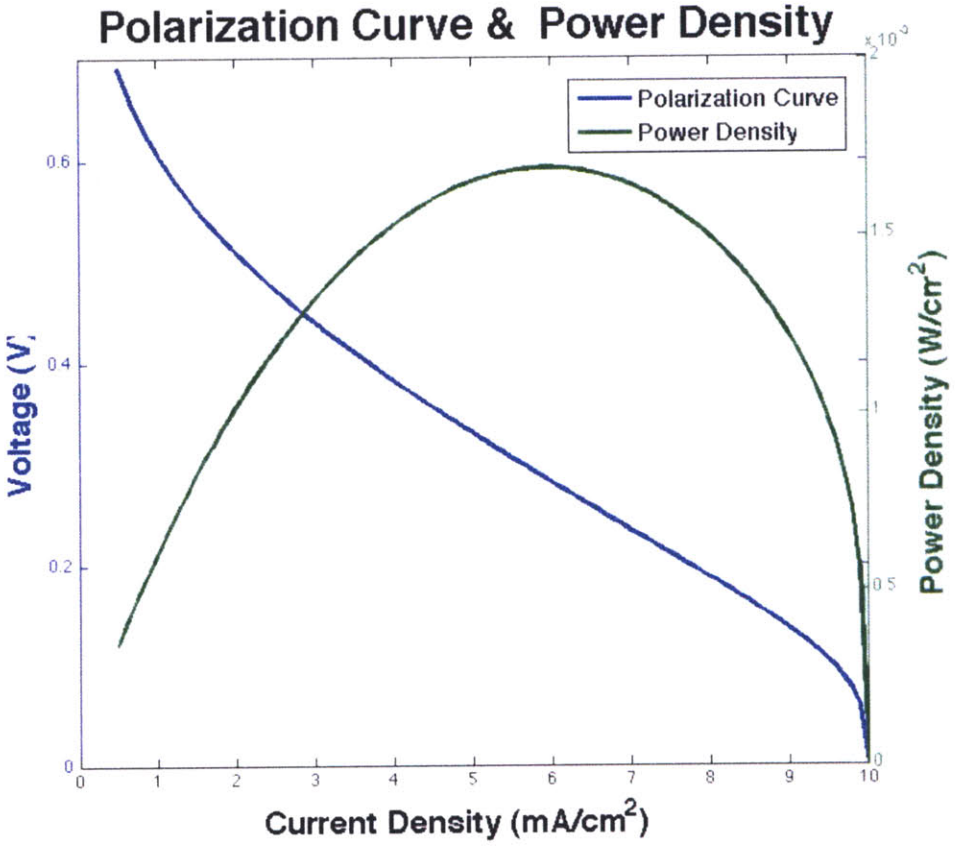


Figure 56: Simulated Polarization curve shown together with power density (P*I). For these plots: L = 4cm; I_{ext} = 0.5 mA/cm²; I_{lim} = 10 mA/cm2; ρ = 1500 Ohm-Cm; R_{ext}=500 Ohms.

Figure 56 shows the simulated polarization curve together with the resuting power density, measured in W/cm² (Note that there is a 10^{-3} at the top of the right Y axis)

of anode surface area. Power density is calculated by multiplying the current density and voltage. The important point about these figures is not the specific values, since these will vary with design variables and constraints. Rather, they demonstrate that the model provides realistic ranges of values for the microbial fuel cell model, given the inputs and constraints we have included.

While power density is a function of the cell voltage and current, we note that the other objective, BOD reduction rate, depends only the current, I. This is because BOD reduction rate is simply a function of how quickly the "fuel" is used, and it therefore does not matter if power is lost or used. One simple way to calculate this rate, then, is to relate the power drawn to the potential energy in the fuel. Acetate has a total power of about 4.0705 Watt-Hours per gram, at a normal OCV (Logan 2008). Multiplying the OCV by the current and dividing by this number (taking into account the right units) will give a rough estimate of the amount of pollutant removed in grams per day.[21] Using this estimate we can compare the power density of the cell versus the BOD reduction rate, as a function of current density for each particular cell (Figure 57).

[21] A more direct way to make this calculation is simply to relate current to the amount of acetate oxidized, given that a mol of acetate produces 8 moles of electrons. This could be added to the model later.

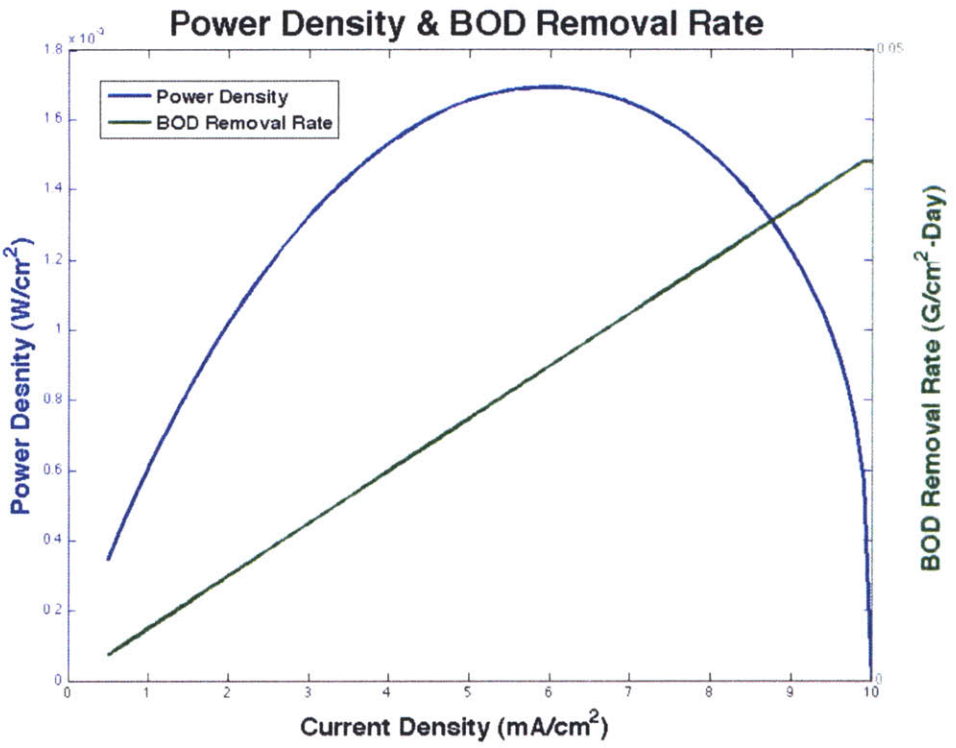


Figure 57: Power density and specific BOD reduction rate as a function of current density.

Figure 57 makes clear that while power density reaches a maximum at mid-range current densities, BOD removal continues to increase until no more current can be generated. The BOD removal rate flattens at high current because in the model we assume that the maximum BOD removal rate cannot exceed the maximum current. At that point it stays constant.

In the objectives defined earlier in the chapter, however, the two goals of BOD removal and Power production were normalized to cost per unit area. This provides the two benefit/cost parameters W/$ and KGBOD/Day$. Using the cost model described above, for this particular fuel cell we can estimate these parameters as a function of current density. Because the specific cost of a given design is a fixed number, this simply involves dividing the results presented above by that number. The Benefit/Cost objective will have a bigger impact on the results where we vary elements of the reactor which impact cost versus benefit differently (such as reactor length).

6.6 The Full Model: Design Variables, Parameters, Objectives

We now summarize the analytical part of the model here. The complete model is provided in the Matlab code in Appendix B. The multi-objective pareto-optimization problem of Microbial Fuel Cell design can now be formulated precisely, as:

Max [P_{cv}, BOD_{cv}]

Where:

$$P_{CV} = \frac{PD_D}{C}$$

$$BOD_{CV} = \frac{BOD_r}{C}$$

S.T.

$$C = a(\frac{100}{L})^{W+1} + C_{Anc}$$

$$P_D = I \times V$$

$$BOD_r = \frac{V_n \times I}{E_{BOD}}$$

$$V_n = V_T - V_B - V_I$$

$$V = V_n - V_{act} - V_{ohm} - V_{conc}$$

$$V_{act} = \frac{RT}{\alpha F}\ln(\frac{I}{i_{ext}})$$

$$V_{ohm} = L(L \times p_e + R_{ext}) \times 10^{-6}$$

$$V_{Conc} = \frac{RT}{nF}\ln(\frac{i_{\lim}}{i_{\lim} - i})$$

$$i_{ext} = 6.8 \cdot 10^{-13} \frac{U_{\max} \times Conc \times T_b \times X \times Yield \times B_{ce} \times F}{K_c + Conc}$$

For modeling purposes, based on the arguments above, we chose a three-variable

design vector: $D = [I, L, i_{ext}]$, where:

I = Operating current density (mA/cm^2)
L = Distance between electrodes (cm)
i_{ext} = exchange current density (mA/cm^2)

Understanding that there are other elements that can be changed, these three design variables correspond to properties of the operations, reactor architecture, and biofilm, respectively. The exchange current, according to our model, corresponds to bacterial properties such as growth rate and columbic efficiency. We therefore make the assumption that the following parameters, among others, can be "designed" using rational genetic engineering or directed evolution. Both have, in face, been attempted with some success.

U_{max}= Maximum growth rate in unit cell mass per day day-1
K_c = Half Saturation constant in mg/L
B_{ce} = Columbic Efficiency of the Bacteria (%)

Varying these latter properties of the microbial biofilm will define the exchange current. Therefore, as described below, we can create bounds on the exchange current without understand the exact way in which the bacterial design variables will actually be changed.

6.7 Limitations to these Modeling Assumptions

A number of limitations to this model should be emphasized. First, the model ignores potential losses accruing from the buildup of Ph in the reactor, and pH gradients across the membrane. These have been shown to cause losses. Second, it is likely that the columbic efficiency of the bacteria is in fact a function of the resistance of the reactor (Logan 2008). Third, the model assumes that bacterial biofilms remain at a fixed density, and ignore differential gradients caused by bulk-liquid dynamics. Finally, the model assumes that we can modulate elements of the biofilm that are, to date, difficult to fully separate. These, and host of other simplifying assumptions suggest that the output of the model will not correspond

perfectly to an actual cell constructed according to Figure 54. These factors might be remedied by incorporating experimental results into the model. However, they are not critical to the actual method described below. The important fact is that a multidisciplinary model can be constructed, and that the relationships between the designs variables identified are correct.

6.8 Quantifying The Implications of Opening and Standardizing

We propose that the multidisciplinary model described above – or any similar pareto-optimization model - can be used to explore the implications associated with standardizing and sharing elements of the design. In particular, opening the design of microbial fuel cells as described here corresponds to the ***Incremental Innovation*** strategy outlined within the broader framework of this thesis. In this strategy, a technology is poorly understood, yet a specific goal can be formulated. The three strategies are reviewed below.

		Suggested Inputs			Suggested Decisions		Likely Result
	Name	**Typical Sponsors**	**System Knowledge**	**Market/Need Knowledge**	**Principal Elements Shared**	**Important Standards**	**Type of Innovation**
1	**Incremental Development**	Users	Low	High	Entire Designs	Measurement, Description	Incremental
1.A	**Database Development**	Users or Developers	Low	High	Data	Measurement, Data Formats	Incremental Pre-Competitive Research
2	**Architectural Development**	Developers	High	Low	Design Patterns, Interfaces	Vertical Standards, Data Formats	Architectural

Figure 58: Review of the three strategies.

Four the microbial fuel cell design case, technology uncertainty stems largely from limiting knowledge of how to engineer the biofilm to increase the exchange current. The fixed needs are the objectives defined above. Within this approach, a basic architecture is standardized and shared, and other elements within the architecture are kept proprietary and competed. The question remains – what should be shared

and kept proprietary, and how might one determine and design standardized elements of the shared section.

The method outlined below helps answer these questions. In particular, we can ask what it would mean to standardize the reactor – as represented by reactor length L – versus the biofilm – as represented by the Exchange Current which is a function of the three underlying design variables. We can also determine the potential implications of standardizing parameters within one or the other.

To do this, we propose a three step method, based on visualizing and extracting relevant subsets within a pareto-set of feasible designs:

1. Create a multi-discipline model
2. Identify physical bounds on relevant design variables and create a pareto plot
3. Identify sub-sets within the pareto plot corresponding to fixed design variables
4. Calculate losses associated with fixing design variables, versus the full pareto set

6.8.1 Step 1: Create a multidisciplinary design model

The first step in the method involves creating the model described above. The second step involves calculating feasible ranges for each design variable. While we know that the reactor spacing, L, can be made as small or large as possible, the exchange current will be limited by physical constraints associated with bacterial kinetics. In particular, K_c will likely vary between 50 and 400 mg/L. For exo-electrogens some have estimated the average to be around 200 mg/L (Logan 2008). The maximum growth rate has been estimated for geobacter to have an upper bound around 8.3 day^{-1}, and we can assume it will not vary beyond between 5 and 10 day^{-1} (Logan 2008). The biofilm thickness can be around 5 microns, or 0.005 cm. The dry weight of exo-electric bacteria has been estimated to be about 3000.

If we assume an input concentration for now of 1000 mg/L, a U_{max} of 8.3 and a half saturation constant of 200, we arrive at an exchange current density of $3.86 * 10^{-4}$

Amps/cm^2. This actually a very reasonable number for an exchange current density of a thick catalyst, given that platinum has an exchange current density of rough 10^-2 Amps/cm^2.

Further, this analysis identifies four variables that can be tweaked to improve the exchange current density: U_{max}, K_c, and B_{ce} and T_b. Let us assume the bacterial columbic efficiency and biofilm thickness are fixed (85% and 0.005 cm, respectively). Varying the maximum growth rate and the half saturation constant between 5 and 10 and 50 and 400 respectively, for an input concentration of 1000 mg/L, creates Figure 59.

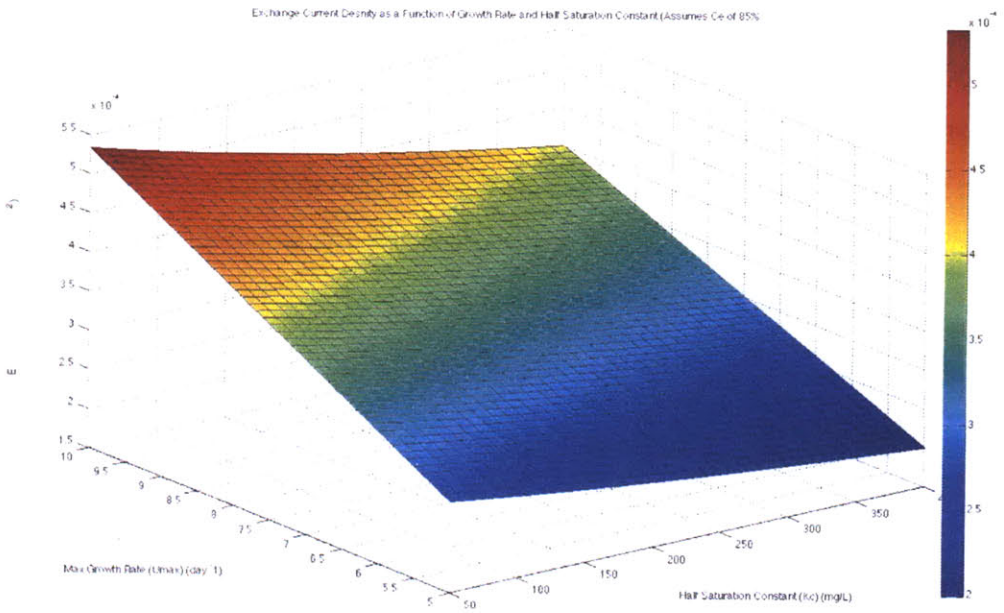


Figure 59: Estimated Exchange Current Density as a Function of Bacterial Kinetics. Y axis is the Half Saturation Constant. X-axis the Max Growth Rate. Z axis is the exchange current density.

The figure suggests that the Exchange Current will likely vary within a fixed range of 0.2 to 0.5 mA/cm^2 for a biofilm that is 5 microns thick. This creates an important, though not fixed, bound on the range of improvement which might be expected through genetic engineering and directed evolution.

Similar bounds can be created around L and operating current, I. In particular, *I* will never be able to be less than zero, and it can never be greater than the limiting

current I_{lim}. L can vary within a much broader range. However, as described below, there are optimal values depending on the other design variables. We can now use these bounds to identify the set of feasible designs.

6.8.2 Step 2: Bound Design Parameters and Create a Pareto Plot

The biofilm modeling section above identified a likely range for i_{ext} of 0.2 to 0.5 mA/cm^2. This was based on varying what will eventually become design variables – B_{ce}, K_c, and U_{max} – within feasible physical bounds. To the extent that the exchange current is a function of these three variables, we can then assume that any future design will vary within these bounds. Because it is likely that a small L is better from the perspective of Ohmic resistance, we vary L within an initial range of 1 to 5 cm.

Holding other elements of the model constant – including input concentration (1000 mg/l), External Resistance (500 Ohms), Limiting Current (10 mA/cm^2), Bacterial Columbic Efficiency (85%), and a number of additional parameters described in the Matlab code in the appendix – we can now identify the performance of a range of designs against the objectives BOD_{cv} and P_{cv}.

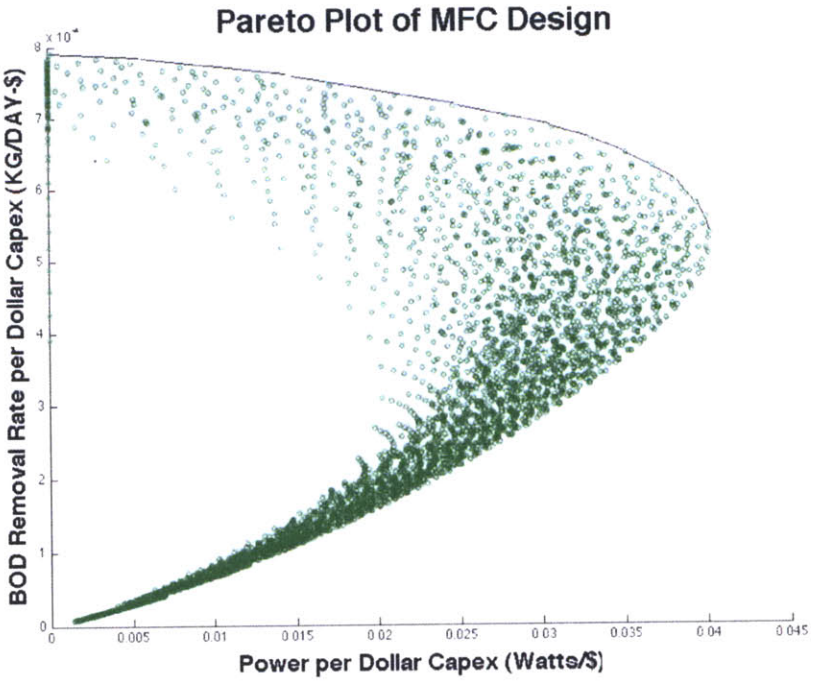


Figure 60: Pareto plot of the Microbial Fuel Cell, representing 2940 Distinct Designs.

Figure 60 is a pareto plot of 2940 MFC designs created by varying the three design variables. The pareto front is delineated with a black line. The designs have a maximum P_{cv} of 0.0402 W/$ which corresponds to $24.87 per Watt. This is, of course too high for electricity generation, but it is only one aspect of the benefit. The figure has a maximum BOD_{cv} of $7.9 * 10^{-4}$ KG/$-Day. This corresponds to $1,265 for every KG/Day reduced. If an industrial customer were to use the system, they would *avoid* tipping fees associated with sending KG of BOD to the local wastewater treatment plant. In particular, one KG remediated costs about $1.50 in tipping fees (NACWA 2005). Therefore it would take 843 days (or 2.3 years) to pay back the cap-ex of $1,265 associated with removing one KG of BOD per day (undiscounted) ignoring, of course, maintenance costs. These numbers are within the range that would be expected for operation MFCs.

6.8.3 Step 3: Partition into Standardized Sub-Sets

Now that a feasible set of designs has been identified within the metric space, we can partition the set of designs into sub-sets corresponding to standardized design variables. This is accomplished simply by partitioning the set of designs associated with different values of design variables. We begin by fixing L and allowing I_{ext} and I to vary continuously.

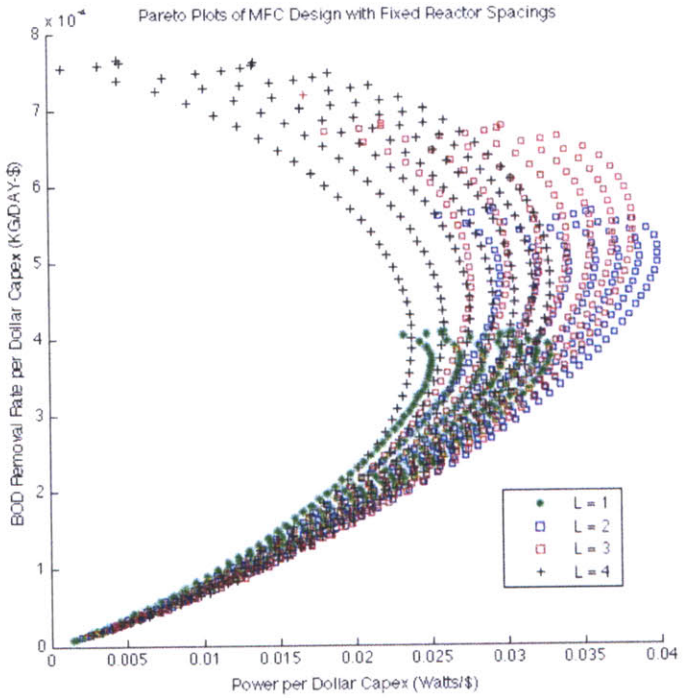


Figure 61: Pareto plot of Microbial Fuel Cell Design. Different colors correspond to different values of L.

Figure 61 was created by varying I and I_{ext} continuously for each L at fixed intervals from 1cm to 4cm. It illustrates the subset of designs corresponding to L's fixed at 1,2,3 and 4 cm. It demonstrates that different objectives will be maximized depending upon which L is chosen, and therefore has important implications for what might be standardized based on the ultimate market objective.

6.8.4 Step 4: Benefit-Cost Implications of Standardization and Opening

The partitioning of the design space and relative performance gains or losses can now be examined quantitatively. Why should different Ls correspond to the optimum for different objectives?

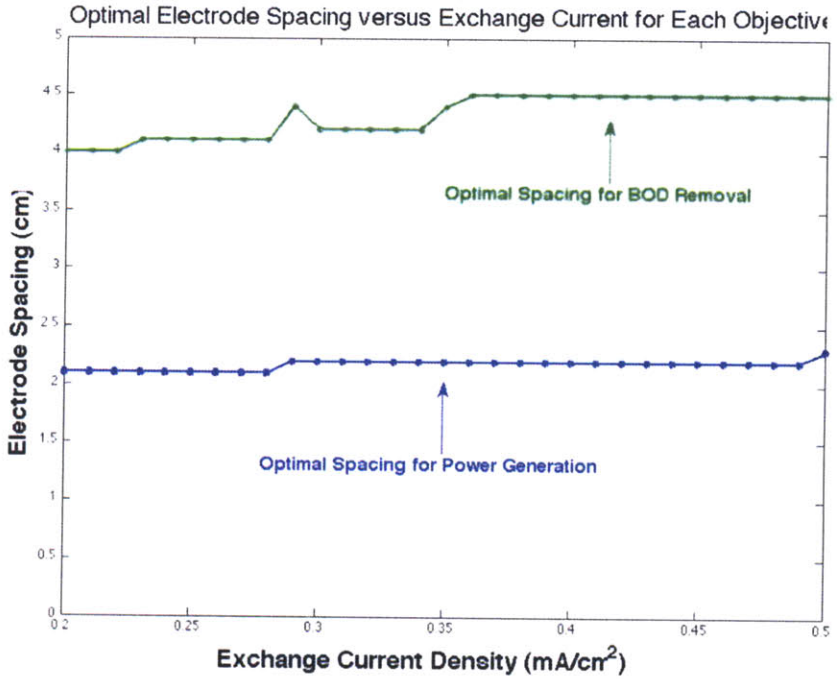


Figure 62: Optimal Reactor Spacing as a function of Exchange Current Density, across all potential operating currents.

Figure 62 was generated by extracting the L for which an optimal P_{cv} or BOD_{cv} is generated across all potential operating currents, I. It shows that the optimal spacing is different depending on whether one wishes to produce power or remove BOD. Further, the optimal values for each objective vary within a fixed range.

This information can be used to evaluate the implications of standardizing around an L which optimizes for power production versus an L which optimizes for BOD removal.

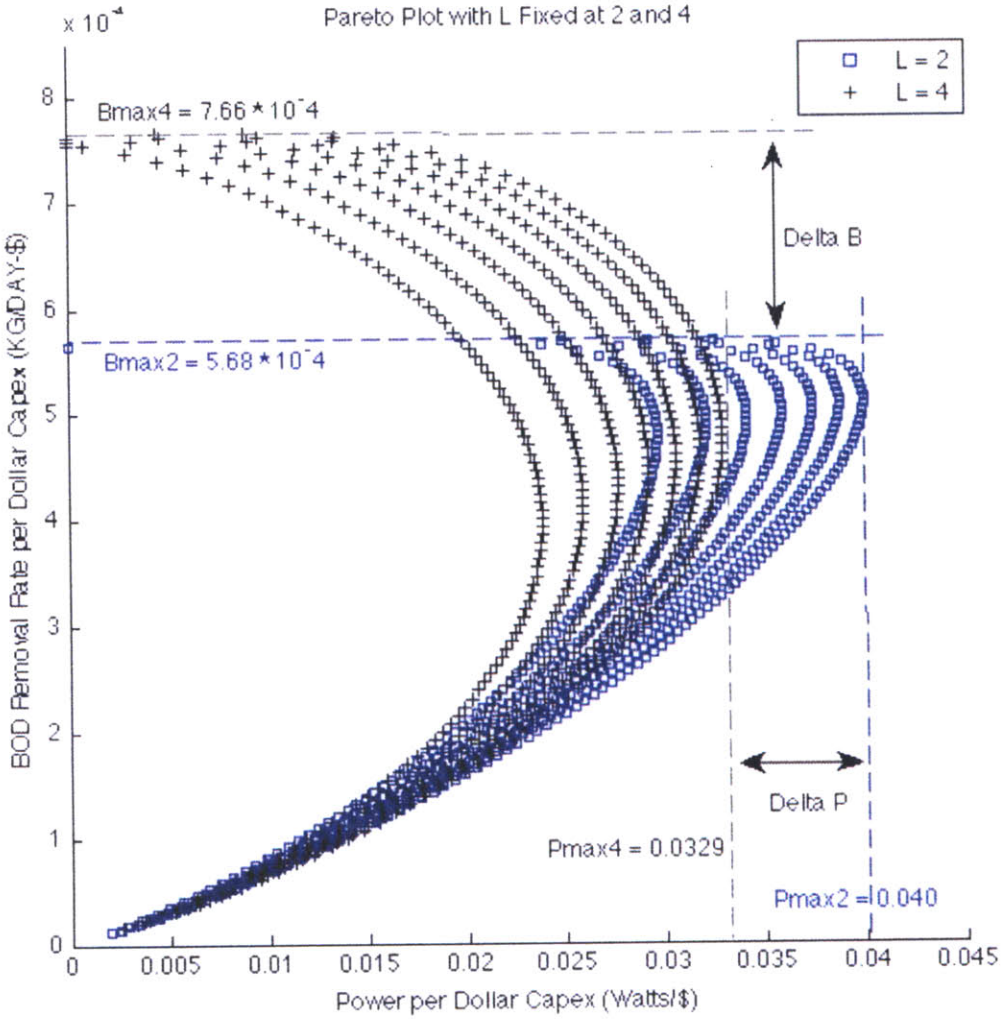


Figure 63: Comparing standardized Ls at 2 and 4 cm.

Figure 63 uses the information generated in Figure 62 to compare subsets created by varying I and I_{ext} around L's, which enable near optimization of P_{cv} and BODcv. Specifically, Figure 62 demonstrates that an L between 2 and 2.2 enables a near optimum for power generation, while an L between 4 and 4.5 enables an optimum for BOD reduction rate, depending on the operating current and the exchange current.

In Figure 63, then, Delta B is defined as the difference between the optimal value of benefit-cost of BOD reduction depending on whether an L of 2 or 4 is chosen. Delta P

is defined as the difference between the optimal value of benefit-cost from power production.

We can now analyze the benefit/cost penalty associated with fixing L in one or the other number.

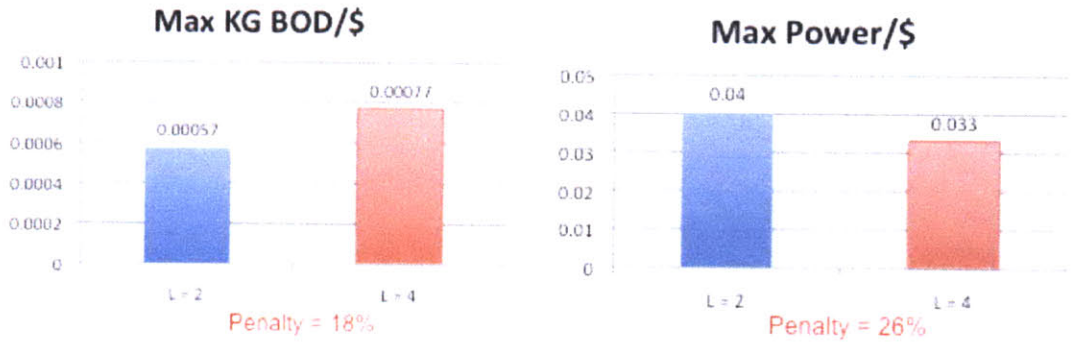


Figure 64: Maximum Power/$ and KG BOD/$ Day for reactor spacing, L, fixed at 2 and 4. The penalties associated with each are listed below in red.

Fixing L at 2 will enable optimal design of Power output, but will cause a ***26%*** drop from the ideal benefit/cost for BOD removal rate. Fixing L at 4 will enable optimal design of BOD reduction rate, but will cause an ***18%*** drop from the ideal benefit/cost for Power.

6.8.5 Step 5: Repeat Analysis with Different Design Variables

The previous analysis was conducted by standardizing L and letting I and I_{ext} vary within their feasible ranges. The same analysis can be performed with the other design variables. For this section we bound L at 5cm. Figure 65 provides the analysis of optimal exchange current (within the feasible bounds) as a function of reactor spacing. It shows that in contrast to the previous case, the optimum for both BOD reduction and power generation is the same, and it is at the highest level of feasible exchange current production.

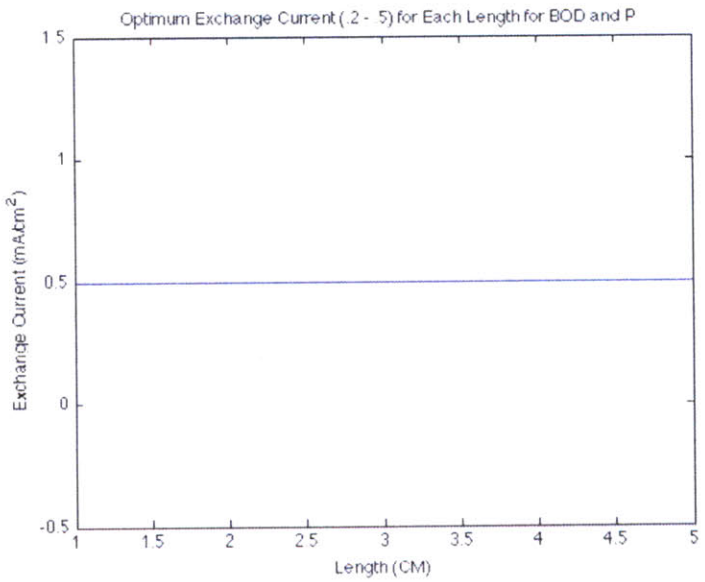


Figure 65: Optimal Exchange current (between .2 and .5) as a function of reactor spacing.

Figure 65 thus demonstrates that fixing the biofilm at a value below the maximum physically allowed will always result in losses to *both* BOD reduction rate and Power Density. What is not yet clear is how much performance of each is impacted. Figure 66 therefore identifies delta P and delta B created by standardizing I_{ext} at the lower and higher values within its feasible range.

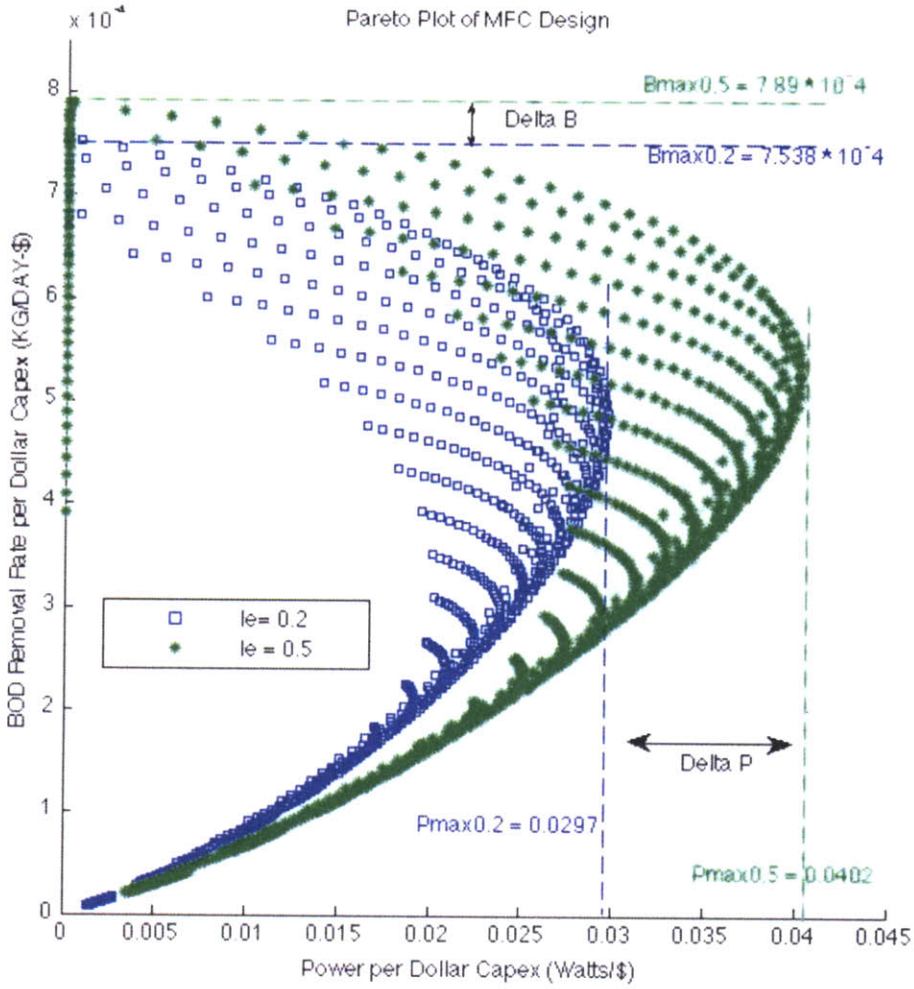


Figure 66: Pareto Plot generated by fixing Exchange Current at 0.2 (blue) and 0.5 (green) and varying L between 1cm and 5cm.

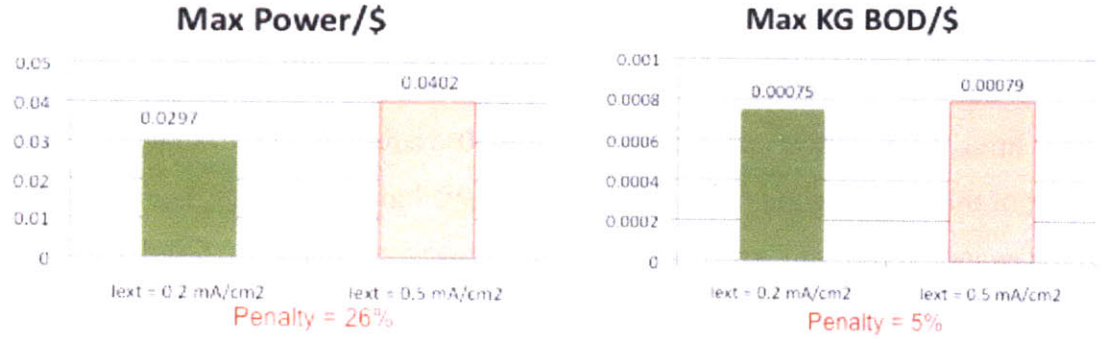


Figure 67: Comparing power production versus water treatment rate for open biology.

The calculation demonstrates that within the feasible range of exchange currents, given the constraints of biology, higher current is always better. For power

production, a 0.2 mA/cm^2 exchange current is 26% worse than 0.5 exchange current. However, for BOD reduction, it is only 5% worse.

The plot thus suggests that standardizing I_{ext} will have a much greater impact on power generation than on BOD reduction rate. While fixing the biofilm impacts both, we lose only 5% versus the potential optimum for water treatment. This must be weighed against the substantial gains in innovation benefits achieved by co-developing products, as described in the framework. Therefore, if the goal is BOD reduction rate, a fixed kind of microbes can be used and participants can compete on the reactor design without significant concern for losing optimality.

6.9 Interpretation of the Results

The method presented here provides information that can be interpreted in different ways depending on sponsor objectives. Within the context of the Incremental Development strategy for open collaborative development articulated previously in this thesis, the modeling results provide concrete suggestions for standardization and opening.

Most broadly, the method enables us to identify the impact of opening one or other aspect of the design in question, as a function of both standardization and objectives. For this particular case, standardizing the reactor spacing can be accomplished in a way which enables an optimum for BOD reduction or Power reduction. Fixing the bacteria – as measured through the exchange current – will always limit both Power production and BOD reduction. This suggests that a sponsor of the open regime may want to create two kinds of standardized reactors – one for water treatment and the other for power production – and encourage developers to compete on the bacteria. The result will be a near optimal reactor spacing for each objective, with the potential for continued improvement in the performance of the biofilm.

Further, performing the method with standardized exchange current demonstrates that the optimal BOD reduction rate does not change that much (5%) between the minimum and maximum feasible exchange currents. This suggests that if a Sponsor's primary goal is water treatment, fixing the bacteria may not have a terribly large impact on future innovation in the space. For these particular sponsors, then, enabling free distribution of bacteria while competing on the reactor elements may in fact yield greater benefits. More analyses would be needed to verify the impact of L on BOD reduction rate given a fixed I_{ext}.

Finally, the method enabled quantification of the negative impact of pre-mature standardization, even given substantial uncertainty about how any of the designs would be realized. This can be used to design standards and determine elements that might be best standardized within the open technology.

6.10 Conclusion and Broader Implications

This chapter develop a quantitative method to explore the implications of opening and standardizing elements of a technology, within the context of the first strategy identified in the broader framework created in this thesis. The method involves created a multidisciplinary model of the technology in question, identifying feasible bounds on input parameters based on physical limitations, performing pareto analysis on fixed subsets of the metric space, and comparing performance losses and optimal values for standardized elements. The method was applied to an integral biotechnology, and used to analyze and craft the strategy for standards design and opening for this technology.

An important attribute of the method involves the fact that the implications of standardization can be analyzed even if elements of the technology are not fully known. By creating a simplified model and identifying feasible bounds on the input variables, the resulting metric space will include designs that are likely feasible but

not yet achievable. This enables the quantification of the impact of standardization without complete articulation of a given design.

More specifically, given competing technological objectives, this method can help:

- Analyze the impact of premature standardization: percent changes from optimum
- Develop metric-specific standards
- Identify what might be opened and left closed

Further research would help identify the limits to the approach, as well as ways in which it can be coupled more directly to the framework and strategies in this thesis. Broader implications resulting from this specific modeling exercise are discussed in the conclusions to this thesis.

7 Conclusion

7.1 Summary

This thesis developed and validated a conceptual framework for the design of Open Collaborative System Development regimes, and explored its application within the domain of biotechnology. The central claim and guiding hypothesis to this work was that the creation of OCSD regimes can be treated as a design problem with the principal goal of creating an environment in which third party developers can innovate. Further, it argued that the methods used to design such OCSD regimes transcend industries, provided that constraints associated with different technologies are adequately considered.

While open design processes have received significant attention in recent years, a number of gaps in the literature prevented the creation of such an encompassing framework. First, there was no clear lexicon with which to described OCSD processes, options, and outcomes. While numerous authors have examined discipline-specific questions associated with opening design and innovation, these elements were not yet synthesized into a coherent conceptual framework for the purpose of creating such a development process. Second, in part because taxonomic categories were not defined, few coherent strategic connections had been identified between even high-level aspects of the regimes. Finally, to date, few studies examined the relationship between systems architecture and other aspects of OCSD processes.

This thesis approached the topic at the level of *systems architecture* and, therefore, utilized and extended the concept of Collaborative Systems as defined by Maier and Rechtin (Maier and Rechtin 2002). It defined "open collaborative system development" (OCSD) as the creation of a *complex system or database in which some or all of the component designs:*

1. Are created by an unrestricted set of potential developers
2. Based on the developers' own initiative (e.g. voluntarily)
3. At least in part for the developers' own reasons

The main challenges associated with applying this method within the domain of biotechnology was defined as follows:

1. The registry of standardized biological parts is an attempt to create an open, collaborative, system development process (OCSD).

2. There are a number of challenges associated with creating OCSDs:
    a. There is no clear lexicon to discuss OCSD
    b. There is no integrated framework to design OCSD
    c. There has been little analysis of how technology type and system architecture relate to OCSD.

The thesis makes the following claim, to be validated through research:

1. An OCSD process can itself be designed.

2. Specifically, OCSD can be treated as a multi-objective design problem where the principal goal is ***to design an environment for third-party innovation***.

3. One can develop an integrated conceptual framework with which to craft strategy and evaluate outcomes for OCSD which includes:
    a. A taxonomy of conceptual building blocks
    b. Options within each taxonomic category
    c. Integrated strategic options

4. The framework can clarify constraints and opportunities in the development of open platforms in synthetic biology.

The thesis employed a pragmatic blend of methods to validate these statements. First, a multidisciplinary literature review enabled the creation of a nascent framework consisting of a set of taxonomic concepts associated OCSDs. Thirteen

examples that fit within the proposed definition of OCSD were then analyzed to further develop the framework and identify correlations between sponsor goals, technological constraints, intellectual property, and type of innovation. These mini-case studies completed the framework and identified three strategies that can be employed by OCSD sponsors, depending on their goals and constraints (Table 15). The framework was then validated through an in-depth case study opening VLSI design. Finally, the framework and strategies were applied within the context of biotechnology. The Architectural Development strategy and the Incremental Development strategy were each applied qualitatively to the design of microbial fuel cell systems. Finally, a simple quantitative method was developed to measure the performance implications of standardization, and applied to the design of microbial fuel cells in the context of the Incremental Development strategy.

As a whole, the research validated the initial claims and produced both general conclusions about OCSDs and specific conclusions about opening in biotech. These, together with the overall contributions of the work are summarized below.

7.2 Generals Conclusions about OCSDs

Chapters 2 through 4 analyzed OCSDs outside biotech, resulting in general conclusions about factors affecting open collaborative systems development. Most broadly, the non-bio case studies confirmed that the basic motivation for opening the design of products and technologies involves acquiring information or, conversely, reducing critical uncertainties. While there are theoretically different ways to achieve this outcome, opening seems to be used by sponsors as a way to increase innovation by reducing transaction costs associated with third party design contributions. These costs could be licensing costs, contract negotiations, patent thickets, or more basic problems associated with conveying and sharing information.

Analysis of thirteen mini-cases demonstrated that sponsors of the open regime were always either users of the technology, or developers of one aspect of the system. These two groups shared the objective of gaining information by increasing third party innovation. However, this objective served different ultimate aims. For users, opening was almost always a means of achieving incremental improvements that increased utility or decreased acquisition costs. For developers, third party innovation was often a way to decrease time to market for complementary economic goods through architectural innovations.

Beyond these basic distinctions, a number of additional correlations were identified or disproved through cross-case analysis. The kind of uncertainty that sponsors attempted to reduce was shown to involve either the system, or the environment, but never both. Further, this constraint correlates to the type of innovation encouraged – whether incremental or architectural. These distinctions, together with a lack of correlation between intellectual property and the OCSD, clarified gaps in the framework and enabled the identification of three basic strategies for developing OCSD.

Table 15: Three strategies associated with OCSD.

		Suggested Inputs			Suggested Decisions		Likely Result
	Name	**Typical Sponsors**	**System Knowledge**	**Market/Need Knowledge**	**Principal Elements Shared**	**Important Standards**	**Type of Innovation**
1	**Incremental Development**	Users	Low	High	Entire Designs	Measurement, Description	Incremental
1.A	**Database Development**	Users or Developers	Low	High	Data	Measurement, Data Formats	Incremental Pre-Competitive Research
2	**Architectural Development**	Developers	High	Low	Design Patterns, Interfaces	Vertical Standards, Data Formats	Architectural

The detailed case study of VLSI design validated the correlations and framework arrived at through the mini-cases. It also enabled a deeper understanding of the way in which technologies can be opened. At a technical level, the VLSI case illustrated

how simplicity of description, reduction of sub-system options, vertical decoupling between design and fabrication, and the portability of design descriptions, can facilitate the second strategy identified through the framework. At a broader level, the opening of VLSI demonstrated the way in which OCSDs entail a different set of goals than closed innovation – it prioritizes speed of innovation and cost of prototyping, often at the expense of performance, per se. These differing goals create different value-networks, as defined by their rank order of metrics. For this reason, they are easily locked-in by socio-cultural and economic. These conclusions underscore the difficulty associated with transitioning from a closed to an open regime, and vice-versa, suggesting that it is best for an OCSD to be formalized within an organization from the start.

7.3 Synthetic Biology Conclusions

Conclusion with respect to employing OCSD in Synthetic Biology fall into four main categories: (1) Performance versus functionality (2) the importance of understanding layering and portability via vertical decoupling (3) management of expertise

At the broadest level, this thesis drew a distinction between creating an OCSD regime that shares full designs in order to incrementally improve *performance* (the Incremental Development strategy) versus those that share design patterns and interface standards in order to diversify end-uses (the Architectural Development strategy). These two goals often, though not always, correlate to whether the technology being designed transforms energy or information. And they correlate to a range of other factors that should be addressed by OCSD designers.

Applications in biotechnology can be *either* energetic and information intensive. To date, most commercial applications, such as fermentation of biofuels or pharmacueticles, are rate-limmited and therefore performance constrained. However, biological designs have the unique ability to evolve after they have been

completed. Therefore, performance losses accrued due to standardization might be partially mitigated via directed evolution. This suggests that the Incremental Development strategy might make the most economic sense in the near-term, and the Architectural Development strategy might be commercially applied later. Yet, for biotechnology, the Architectural Development strategy might be applied sooner than in many other industries. In this context, "sooner" means that it might be applied before the underlying systems are fully, rationally, understood.

This thesis also suggests that application of either strategy will benefit from a clear understanding of *layering* in complex systems. Most OCSD regimes examined in this thesis shared one or more entire layers of abstraction, or simply shared open, vertical interface standards. Within the context of biotechnology, this suggests that one might consider creating a Registry, for example, in which the parts and device level are freely shared, but systems can be fully protected.

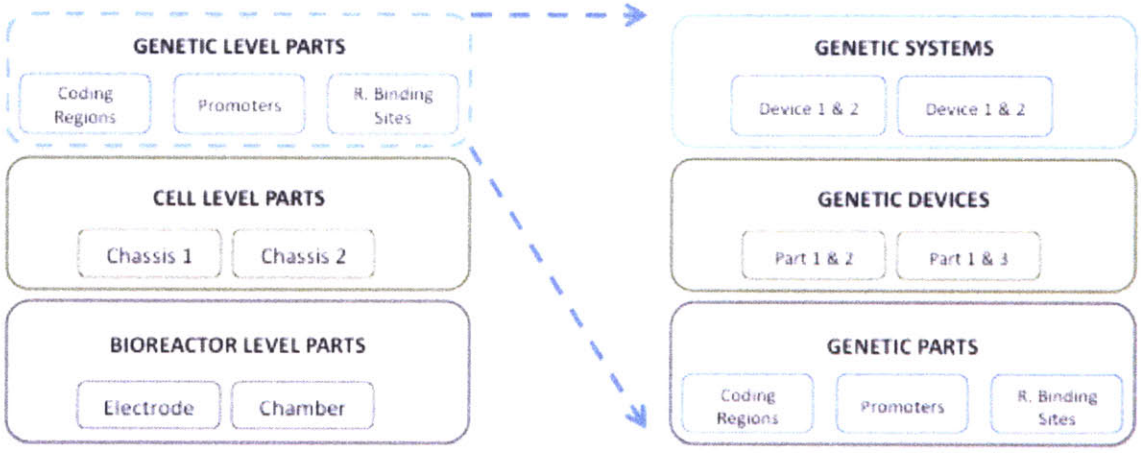


Figure 68: Two nested abstraction hierarchies in Synthetic Biology. The bioreactor here includes "electrodes" because it is part of a microbial fuel cell. Different applications will have different bioreactor parts.

However, to accomplish this in a commercial context, two factors must be considered. First, as noted in Chapter 4, the Architectural Innovation strategy is dependant on achieving portability across a given layer of abstraction. Second, as noted in Chapter 5, the distinction between DNA parts, devices, and systems, is actually part of a broader abstraction hierarchy that includes bioreactors, cells, and DNA (Figure 76). This research suggests that creating open, vertical standards

between these broader levels will likely impact portability to a greater extent than focusing on horizontal standards between elements within a given level. To be sure, horizonal interface standards facilitate vertical decoupling. However, as demonstrated in Chapters 4 and 5, they are not mandatory.

The three strategies identified in this thesis also place constraints on the way in a biotechnonological OCSD process might be designed. In particular, the research suggested that most sustainable OCSDs have the goal of reducing end-use uncertainty, or system uncertainty, but not both or neither. The Registry of Standardized Biological Parts, as it stands, has no clear end-use goal (other than to facilitate biological engineering), yet there is also a high degree of uncertainty associated with the underlying systems. If the goal is a sustainable academic *and* economic initiative, it might help to fix end-goals. It may be useful to create registry sections, or new registries, dedicated to individual goals such as producing bio-fuels, sensors, medical solutions, or the like. While this partitioning risks fracturing community knowledge in the near term, it will likely result in faster accumulation of useful biological parts which can, one day, be combined to create a more generic registry.

The case studies and framework provide additional suggestions if the end goal is Architectural Development of genetic circuits. First, biologists may consider reducing the number of parts in each basic category to the extent feasible. Both the number of elements *for each function* and, the number of functions *at each layer of abstraction* should be restricted. This will decrease the amount of learning needed to design new biological systems. More importantly, it will limit the potential interaction effects between parts and between the design and the environment, facilitating rapid-prototyping. As with VLSI (chapter 4), limiting the range of parts and functions at each layer of abstraction may reduce the number of iterations needed between different layers in the design problem.

At a more general level, the VLSI case in particular illustrates the double-edge sword of expertise and complexity. In the late 1970s, both large companies and individual LSI circuit designers viewed deep expertise in the design of integrated circuits as competitive assets. Xerox viewed its "standing army" of designers as a potent competitive weapon, enabling it to develop highly sophisticated, hierarchically decomposed LSI designs. This view was re-enforced by the actual division of labor within the company, through which deep expertise was rewarded with status. Ironically, these very perceived sources of sophistication prohibited practitioners from fully appreciating the potential gains in design productivity that might accrue from simplifying and sharing designs. The approach to opening VLSI advocated by Mead and Conway forsook the goals of performance and sophistication in the name of rapid prototyping and ease of learning. Even if kept fully secret, this basic approach may have helped large companies reduce prototyping costs. Yet, the value network in which the companies operated, and the resulting incentives structure within the organizations, seemed to lead people to discount the importance of the objective in the first place. At the very least, it leads to a heavy discounting of the potential benefits.

The story has direct relevance within the domain of biotechnology, where deep expertise and highly specific knowledge are amply rewarded with status and prestige. This is, of course, to be expected given the tremendous complexity of biological systems. But the story of VLSI suggests that the limitations and constraints created by the expert mindset are not always obvious to those within the system. For example, while it may be obvious to some that development costs might drop by simplifying descriptions, decoupling systems, and facilitating learning. These goals may be unatenable if carried out by practioners whose approach is seeped in the deep exerptise associated with research. Worse yet, some might finds that the goals achieved by making these gains may themselves be viewed as less important by experts in the field. The effect may be more subtle than many assume, affecting the field through indirect factors such as the content of

coursework, and the criteria for promotion of mid-level managers in biotech companies.

These four observations – the need for specific end-goals, the need for a smaller set of design options, the need to emphasize vertical portability within the context of biology, and the potential pitfalls of over-specialization – are, in fact, related. For example, a simpler set of design alternatives may well constrain the kind of host organisms that can be used, which will also faciliate portability while limitting the range of end-applications. As with all engineering challenges, design of an OCSD process entails understood the trade-offs.

Finally the quantitative model developed in chapter six could have implications for the development of at least some kinds of standards in the biological domain. When applied to microbial fuel cells with a fixed end-goal, the model suggested that it was in fact the reactors, and not the biology, which should be opened and standardized. This was due, in part, to the relationship between the reactor elements and the objective functions specific to that problem. However, the basic method proposed could be applied in other contexts. It would be useful to determine whether this is often the case.

7.4 Thesis Contributions

In short, this thesis has made the following contributions:

1. Validation that open, collaborative system development can be treated as a design problem with the goal of developing an environment in which third parties innovate.
2. Development of a multidisciplinary taxonomy for OCSD.
3. Creation of a framework to link taxonomic concepts, and validation of this framework.
4. Identification of three strategies that recur within OCSD.
5. Description of how the framework and strategies might be applied within biotechnology.
6. Development of a quantitative method for the analysis of the implications of standardization and opening, in the presence of technological uncertainty.

7.5 Future Work

There are a number of limitations to this research that could be addressed through future work. First, the high level of analysis associated with the mini case studies limited the depth of conclusions. The correlations identified were consistent, but could have been more rigorously defined. For example, modularity of the technology was binned according to judgment of the author. A more detailed approach would have been to create a design structure matrix for each technology, and use this to measure modularity according to well known clustering algorithms. However, this level of detail would have reduced the breadth of the cases reviewed and limited the thesis realistically to the case studies in chapter three.

More generally, the underlying premise of this thesis – that OCSD can be treated as a design problem independent of industry – lead to a cross-industry case selection. In addition to the correlations identified in chapter three and four, there are significant differences between the industries and technologies analyzed. These differences were highlighted in the thesis, but not explored in-depth given the thesis goals. Further work might further explain and define the differences between the cases.

If the case-study work required an understanding of the dangers of over-generalization, the modeling work in chapter 6 underscores the limitations associated with specificity. The proposed method to quantify the impact of standardization and opening on potential technical performance is demonstrated through a very specific technology. It is not clear that the method would be easily transferred to other biotechnologies, or other industries. This also merits further work. For this technology and within the context of the strategic framework, the method suggested that opening the reactor architecture would have a better impact on innovation than opening the biological elements of the design. However, this conclusion might well change for different biotechnologies.

As a general matter, the limitations of both methodology and conclusions in this thesis are, to a certain extant, illustrative of the current limitations of multidisciplinary work in Engineering Systems. By definition, Engineering Systems seeks knowledge that spans both engineering design and sociological knowledge claims. The methods for acquiring and interpreting information in each domain are very different. Therefore, scholars within ESD currently have a choice. They can utilize methods and ask questions that fit fully within existing disciplinary boundaries, such as engineering design, management, or political science, or they can combine methods and ask questions that span boundaries. However, because the latter approach applies multiple methods to different kinds of questions, the results cannot conform to standards created within each discipline. This approach thus runs the risk of appearing incomplete when interpreted through the lens of existing disciplines. Hopefully, as the field matures and the audience with ESD-specific background grows, these kinds of challenges will fade in the light of the relevance of the research.

7.6 Broader Implications and Future Work

Beyond the specific conclusions and contributions associated with this work, the process of examining these topics raised three basic themes that the author believes are worthy of continued research, and may have significant practical implications. The first is the basic possibility of treating Open, Collaborative Development itself as a design problem. The second is the related implication that there is increasingly a shift in industry, enabled in part by communication technologies and flexible manufacturing, from an era of mass production to an era of mass innovation. The third involves the potential for synthetic biology to merge aspects of information economics with the production of physical goods.

This thesis provided preliminary work towards the idea that one can design open innovation environments. The complexity of the task, however, became apparent early in the endeavor, and it is likely that an entire research program could be

designed around the goal. For example, though high-level distinctions and strategies were identified through the case-study work, these were not likely exhaustive. The link between, among other factors, legal regime, incentives, and technological architecture, and standards should be explored in more depth. Ideally this, together with broader approaches to simulation, might enable outcomes to be measured more specifically than in this thesis.

A recurring theme throughout this research was the importance of innovation in and of itself as a measure of competitive advantage. Though this can be overstated – especially in the management literature – it takes on concrete meaning when one considers that the average "generation" of some technologies is only a few years, and the time for new generations often drops as an industry matures. Within this context decreasing the cost of new versions can be as important as decreasing the cost of production. The latter is typically associated with mass manufacturing, and the standardization of production. Yet, this research suggests that trade-offs associated with mass-production – higher capital costs versus lower unit costs – have direct analogies in the context of "mass innovation." The figure below was created to highlight the similarities.

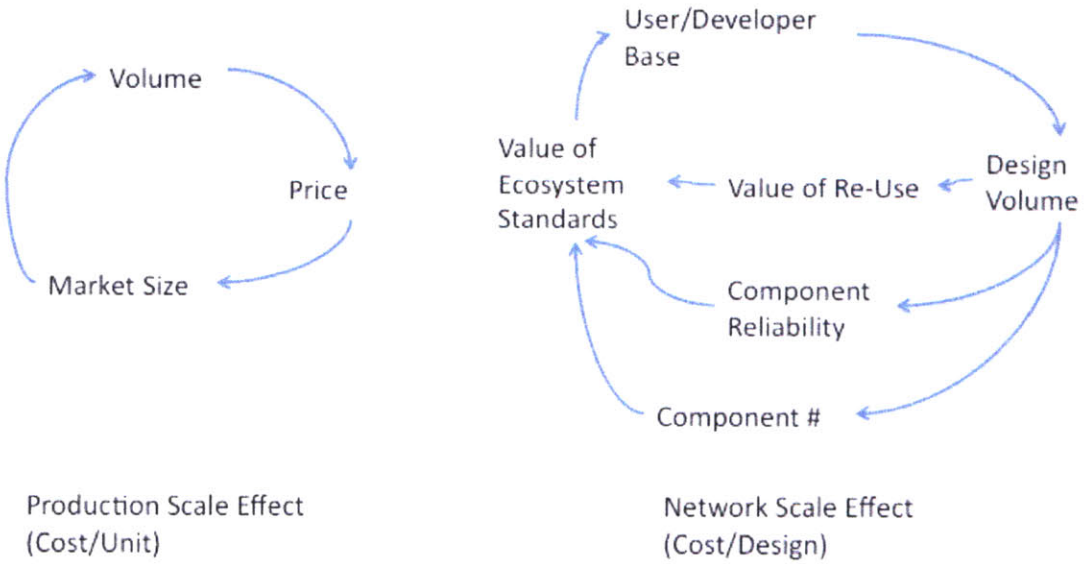


Figure 69: Notional model of mass production and "mass innovation."

The cost advantages of mass production result from scale effects associated with amortizing up-front capital costs across large numbers of units. In much the same way, scale effects become important in a mass-innovation regimes. An up-front investment in standard interfaces, descriptions and languages will enable individual components to be re-used, thus amortizing the cost of a "design infrastructure" across multiple design generations. The benefits of scale in this case stem from network effects associated with the re-using components and sharing descriptions across a community or industry.

The concept of "mass innovation" raises interesting possibilities with respect to openness. One might ask, for example, why the benefits associated with mass-innovation cannot be instantiated within a single company. Within the context of biotechnology or software, this does seem inherently feasible. Microsoft, for example, encourages the re-use of design patterns within the company's "ecosystem." Similarly Amyris, a synthetic biology company, is purported to re-use some of its genetic parts in different systems. However, the implication that network effects increase the benefit of mass-innovation suggests that where mass innovation is possible, openness is likely to follow. This is because the pre-requisites for open-collaborative system design and for mass-innovation may be very similar. These are, of course hypotheses that could be confirmed or refuted with further work.

Finally, a number of themes involving the intersection of economics and biotechnology that emerged through working on this thesis merit much deeper analysis. In particular, biotechnology produces material goods, yet it shares attributes of information in that it is expensive to produce but cheap to reproduce. This observation has been made before, and it can be debated, but it takes on special importance in the context of open design. Economists emphasize that most goods should eventually trade at the marginal cost of production (Shapiro and Varian 1999). If synthetic biology is successful – and DNA design is decoupled from chassis design or chassis design decoupled from reactors – the marginal cost of production should eventually become very low. Combined with the distribution of design

knowledge, low marginal production costs in biotech have the potential to alter the way in which industrialized economies produce basic physical goods.

The question concerning the extent to which Synthetic Biology will merge with the economics of information with those of production would, therefore, appear to be worth closer examination. To the extent that this occurs, understanding open design within the industry is all the more important.

8 Appendix A: Case Studies Not Presented in Ch 3

This appendix presents and describes the additional mini-cases analyzed for chapter 3.

8.1 Bessemer Patent Pool 1866 - 1877

8.1.1 Background

Allen and subsequent authors cite the creation of a Bessemer steel patent pool as another example of collective invention. It is included here because it one of the common examples cited in this literature – yet it differs from other examples because it involves the pooling and licensing of patents. Eventually the pool was closed to outside licensees.

Henry Bessemer took out a patent for his basic process for converting Iron to Steel in 1856. A number of conflicting claims soon arose with respect the basic technology, precipitating a legal battle between two camps. These conflicting claims prompted an engineer and consultant, Alexander Holley, to organize a patent pool in 1866 in the United States (Meyer 2003). The pool licensed the technology to user-producers such as railroad companies. The Bessemer patent pool continued to function after the initial Bessemer patent lapsed. By 1877 a series of mergers in the industry ended the practice of accepting licensees, though some were able to join at significant higher price (Meyer 2003).

Importantly, access to this pool was not free. A license to all patents in the pool was $5000 dollars initially and five dollars per ton of steel produced - a significant amount in the late 1800s (Whitten, Whitten et al. 2005). In this way, the case differs from others in this chapter. Yet, the licensing terms did stipulate that those in the pool should not only have access not only to the patents, but must share knowledge about their application and operation (Allen). Members could send two employees

to firms already using the Bessemer process for the purpose of knowledge transfer and would receive regular briefing and newsletters written by Holley (Meyer 2003; Whitten, Whitten et al. 2005). In exchange, members were obliged to open their plants to others, and contribute knowledge to the basic pool.

8.1.2 Organizational and Economic Context

The Bessemer Patent pool arose through an out-of court settlement for potentially intractable, conflicting patent claims. This was between a group led by Holley out of Troy, New York, and a group led by Businessman/Investor Ebar Brock Ward (Thomas 1995). While the former had licensed Bessemer's patents including important "tilting" technology, the latter owned elements of the post-treatment process (adding manganese) and air blowing. It was only through three-years of negotiation beginning in 1866 that a solution was resolved in the form of what would become the Bessemer Association – a patent pool in which the Troy group received 70% of the profits and the Ward group received 30% (Thomas 1995). The pool then licensed the technology to eleven rail mills between 1866 and 1877 – or users of the technology. After 1877 it was closed to outsiders. In short, the Bessemer case differs from Allen and Von Hippel's discussion of "free revealing" of user information. It is rather an example of *Open Innovation* through patent licensing as defined by Chesbrough.

Still, some important factors can be identified more generally. First, unlike the Allen case, the firms in question were not bounded by geography. They spanned the entire United States, as it then existed. Unlike the furnace case, sponsoring firms did not also own complementary mining assets. Licensees of the technology did. Finally, these firms were in competition with rival steel processing technologies. In particular, the "open hearth" method had the advantage of enabling extra phosphorous concentration in the input process, thus broadening the range of inputs with equivalent or even superior and quality (Whitten, Whitten et al. 2005). Therefore, while the common pool did not compete against competing regions, it certainly competed against alternative technological solutions.

8.1.3 System Architecture

Bessemer converters are highly integral technologies. They consist of large vats through which molten pig iron and air can flow, with elements of the lining, geometry, inlets, and outlets all affecting steel composition.

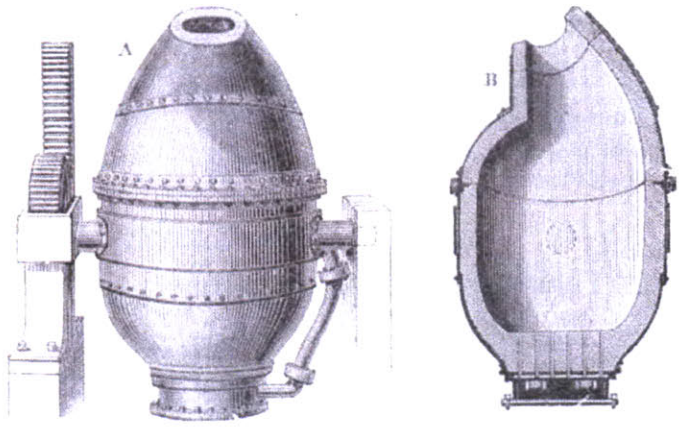


Figure 70: Bessemer Converters. (Courtesy Wikipedia)

The converters are designed to purify pig-iron into steel. The process requires mechanisms for blasting air into the vat, but also for tiling the vat, and re-introducing elements into the iron to aid with malleability. One might argue that it was, in part, the integrality of these elements that lead to the possibility of a patent thicket.

8.1.4 Intellectual Property

The important elements of the legal context were described in the section on organization and economics. In particular, the technology was patented through multiple patents. This created a patent-thicket and impetus to pool patents. Once a pool was formed, the technology was licensed to user-firms who then had the right and responsibility to exchange best practices. In 1877 the number of licensing firms was curtailed.

8.1.5 Case Summary

- **Stakeholders**
  - **Sponsors:** Steel firms
  - **Developers:** Steel firms
  - **Users:** Railroad Firms
- **Objectives and Strategies**

    - **Sponsors:** Avoid patent thicket
    - **Developers:** License to railroad firms
- **Technology**
    - **What was shared:** Component Patents and Best Practices
    - **Modularity of Architecture:** Low
    - **Standards Used:** N/A
- **Intellectual Property**
    - **Architecture:** Patent Pool
    - **Parts:** Patent Pool
    - **Standards:** N/A
- **Constraints**
    - **Value Chain:** Integrated
    - **System Knowledge:** Low
    - **User-Need Knowledge:** Hi/Commodity

8.2 Cornish Steam Engines

8.2.1 Background

Nuvolari has identified evidence of collective invention in Cornish pumping engines (Nuvolari 2004). In the late 18th century in England, Mining entrepreneurs ("mine captains") needed to remove water from mines at low cost. Simple, inefficient steam engines were used early in the century. In mid-century Matthew Boulton and James Watt invented an engine which utilized a separate condenser for improved efficiency. In 1776 the first Boulton & Watt engine was installed in Cornwall. Due to its clear superiority to existing engines and the breadth of their patent, Boulton & Watt were able to charged high royalties for use of the engine.

The onerous licensing fees resulted in the creation of several "pirated" version of their engine in the district (Nuvolari and Verspagen 2007). Nuvolari documents subsequent battles between Boulton & Watt and local engine entrepreneurs who sold engines that infringed on the patent. Nuvolari notes: "Boulton & Watt, with their legal victory (pursued with relentless determination), completely alienated any residual sympathy towards them [in Cornwall]" (P 169). The net result was that upon expiration of the Boulton & Watt patent in 1800, not a single engine was purchased from the company.

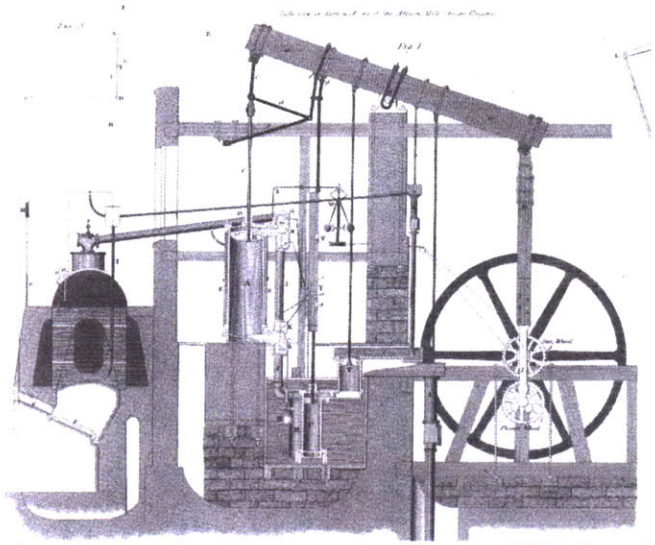


Figure 71: Engineering Drawing of one example of Boulton & Watt's engine. (source: http://www.uh.edu/engines/)

The expiration of the broad Boulton & Watt patent created a space in which numerous engine-men could tinker to improve the existing architecture. In 1811, a few years after the lapse of the patent, a group of mine captains decided to openly publish these improvements. They asked a highly respected engineer Joel Lean to create a monthly publication called the *Lean Engine Report* that would make public engine information. Their goal was explicit: (1) The rapid identification and diffusion of best practices and (2) creation of competition among engineers entrusted with different engines in order to increase the rate of progress (Nuvolari and Verspagen 2007).

The distribution of this knowledge corresponded with the use of high-pressure steam in the engines. Over the ensuing forty years, techniques for utilizing high-pressure steam expansion were perfected and reported in the Lean Engine Reporter, leading to a dramatic improvement in the performance of Cornish Steam Engines (measured in *duty* or unit of water raised one foot per unit of coal consumed) Figure 72. Nuvloari notes that rapid increase coincides with Lean engine reporter and decline seems to coincide with depression after 1850 where copper and tin prices collapse and so experimentation drops.

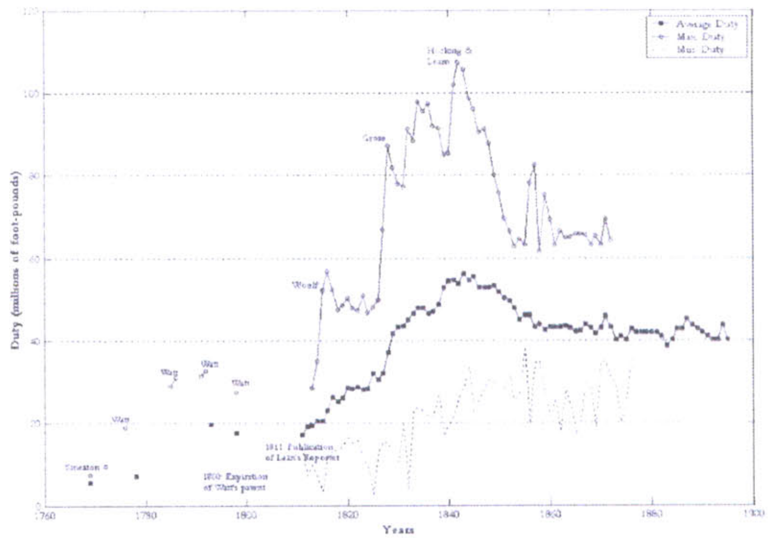


Figure 72: Cornish Steam Engine Performance. Open design began in 1811. (Source: Nuvlari, 2004)

8.2.2 Organizational and Economic Context

The Cornish engine case bears similarity to Allen's furnace case at an organizational level. Many of the firms and entrepreneurs sponsoring the construction of engines were also owners of the coalmines, made more valuable by efficient pumping. The sponsors of the regime were therefore user-innovators, as described by Von Hippel. They own complementary assets and treated the engine technology as an input to production. The output was sold to a "world market" making the firms price takers.

Further, it should be noted that the particularities of Cornwall made efficiency of coal use very important. Nuvolari writes: "In comparison to other counties, Cornwall was characterized by a relatively high price for coal which was imported from Wales by sea." (Nuvolari and Verspagen 2007). This suggests that, in addition to the high rents charged by Boulton & Watt, mine owners had a greater incentive than in many other counties to lower coal consumption.

Importantly, tinkering and patenting were not discouraged among the engine-men. Therefore, two important sets of incentives were created by publication in the Lean Engine Reporter. First, the reputation of the engineers could be enhanced by publishing improvements and high performance. Second, the reporter could serve

as a marketing tool if new concepts were developed. Nuvolari suggests that the latter may have in fact occurred.

8.2.3 System Architecture

The goal for these engines was of course purely performance. This was measured in terms duty – defined as number of pound-feet of water lifted using a fixed amount of coal. To calibrate these reading with design, the reporter included the following technical information for each engine: (1) Diameter of the engine cylinder (2) the length of each stroke (3) the number of pumps connected to the cylinder (4) the diameter of each pump. It also included operations and performance information such as the load on the cylinders, the number of strokes per minute, the amount of water lifted. Together, this information could be used to correlate design and operations to performance.

As in the other industrial-era cases, the designs in question were largely integral and tightly coupled. Designers lacked sufficient theoretical knowledge to adequately guide optimization. Nuvloari writes: "Vincenti suggests that engineers tend to make use of systematic data collection to bypass the absence of an adequate theoretical understanding of the operative principles of a technology. This was exactly the situation in early nineteenth century steam power technology when no fully fledged understanding of the working of the steam engine was available." (Nuvolari 2004)

Further, the use or operation of these engines was poorly developed. The lean engine reporter had the goal of encouraging best practices for operation. This is because powering the engines required skill and a lot of tacit knowledge.

8.2.4 Intellectual Property

Intellectual property and patents clearly play an important role in this case. As in the furnace case, a specific architecture was not patentable – in this case because the watt patent had lapsed. Further, the most important aspect of experimentation – the use of high-pressure condensers – was mentioned in Watt's original patent and thus was clearly not patentable either. Yet, as in the Allen case, elements of this open

architecture were patented and promoted. The Lean Engine report was therefore as much a method to share "best-practices" as it was a method to collectively design systems. Patenting of components was very much a part of the culture and was not dissuaded through licenses, social norms, or other mechanisms.

8.2.5 Case Summary

- **Stakeholders**
  - **Sponsors:** Mining Firms
  - **Developers:** Consultants/Engineers
  - **Users:** Mining Firms
- **Objectives & Strategies**
  - **Sponsors:** Lower input costs
  - **Developers:** Sell design and maintenance services; Reputation
- **Technology**
  - **What was shared:** Design and Performance Data
  - **Modularity of Architecture:** Low
  - **Standards Used:** Measurement only
- **Intellectual Property**
  - **System:** Public Domain
  - **Parts:** Some parts patented
  - **Standards:** N/A
- **Constraints**
  - **Value Chain:** Vertically Integrated
  - **System Knowledge:** Low
  - **User-Need Knowledge:** High

8.3 The SNP Consortium and the Human Genome Project

8.3.1 Background

In the 1999 a number of normally highly competitive Pharmaceutical firms surprised many by creating a consortium to share proprietary information about the human genome and place it in the public domain. The public database, made in collaboration with IT firms and non-profit universities, included information about genetic differences between individuals called "single nucleotide polymorphisms" or SNPs. SNPs are the genetic letters that vary between individuals resulting in morphological or behavioral variety (as opposed to genetic differences which separate our specifies from others). SNPs, individually or collectively, can therefore also indicate the presence and cures for disease. The SNP map can be used to identify promising drug targets.

Information developed by researching SNPs could therefore be considered highly valuable and proprietary. Yet, the SNP consortium members collectively contributed over $50 million of dollars to map 1.8 miilion SNPs and place them in the public domain[22] (Tapscott and Williams 2006).

8.3.2 Organizational and Economic Context

The Sponsors of this open regime included for-profit Pharmaceutical companies, IT companies, and non-profit institutions. Their stated goal included providing a shared map for both scientific and industrial use. The website for the SNP consortium emphasizes that SNPs are a map that would form a pre-competitive research tool to the benefit of large companies and society (ORNL 2009). But scientific sponsorship alone surely does not fully justify the expenditures and break with traditional proprietary practices.

Some authors have identified several organizational, legal, and technical factors which motivated the collaboration by Pharmaceutical companies. From an organizational perspective, it is important to know that Pharma companies view SNP information as inputs to their core-competence and production. That is, while such companies dedicate significant resources to "discovery" processes which can identify lead chemicals or targets for drugs, they also add tremendous value validating, testing, and commercializing these leads. The biotech industry is characterized by many small firms which can provide "lead identification" or "target discovery" but do not have the resources for full-scale characterization and clinical trail. Thus publishing results avoided a risk of significant wrangling among the large Pharma companies and between them and smaller biotech firms with claims on various SNPS. Tapscott and Williams note: "Like many pharmaceutical companies, Merck sees gene sequences as inputs rather than end products....by placing gene sequences in the public domain, Merck preempted the ability of biotech firms to encumber one of its key inputs with licensing fees." (Tapscott and Williams 2006)

[22] Consortium members were: APBiotech, AstraZeneca Group PLC, Aventis, Bayer Group AG, Bristol-Myers Squibb Co., F. Hoffmann-La Roche, Glaxo Wellcome PLC, IBM, Motorola, Novartis AG, Pfizer Inc., Searle, and SmithKline Beecham PLC,The Wellcome Trust.

8.3.3 System Architecture

If the organizational structure of the biotech industry created IP concerns that gave impetus to collaborative public disclosure, why has this only happened in rare cases like SNP? An answer to this depends in part on the technological factors in question. In particular, uncertainty played a very important role. Not only were there concerns about individual SNPs – but the unknown relationships between SNPs created the possibility of untold conflicts between SNP claims.

More specifically, two kinds of technological information are relevant to this case. The first is the biological data itself. And the second is the database technologies used to share the information. The former is of principal concern here, as it motivated the collaboration. Before addressing uncertainty, it is important to note that the task-structure associated with creating the database was modular. While it is somewhat ambiguous to speak of the "modularity" of a data set, we can safely state that the activities of the consortium members were fairly independent. Developing SNP data could be done without collaboration with other members, and duplication of data-points could only help, not hurt, accuracy.

Despite the modularity of tasks, significant uncertainty clouded both the value of individual SNPs and their interrelations. First, because the information being developed was based on extremely novel gene sequencing technologies, there were important outstanding questions about scientific validity. Member firms therefore had to be concerned that even if they identified an important set of SNPs, drug-approval might be delayed by scientific questions about how those SNPs were found. As one author writes: "Consortium members hope that it will be easier to win approval if the tests use markers that are in the public domain, and are therefore subject to challenge and validation by the scientific community." (Eisenberg 2000) Underlying questions about the validity of research results create an impetus for peer review.

Uncertainty also existed about the ultimate value of any given SNP and the relationship between various SNPs. For this reason, though the task structure was highly modular, the value of the technological artifact being created was uncertain, even to the developers. This uncertainty increased the probability of conflicting IP claims. Thus, unlike many potential targets in the Pharmaceutical industry, the potential for conflicting claims and IP wrangling was impossible to predict and dependant on complementary activities of other biotechs and pharmaceutical companies – unless, of course, the database was put in the public domain.

8.3.4 Intellectual Property

Many of the salient IP questions about this case have been discussed in the previous sections. Architects of the program note: "the number of SNPs that (1) enter the public domain at the earliest possible date and (2) to be free of third-party encumbrances such that the map can be used by all without financial or other IP obligations." (Eisenberg 2000) The final product was therefore a free and open database which could be used by scientific researchers and companies alike.

A broader point might here be made about the concerns about intellectual property within the biotechnology industry. The race to characterize human genetic differences for profit raises a host of ethical question, made particularly acute by the notion that one company might "own" parts of the human genetic code. Therefore, the efforts of the SNP consortium, while explainable based on technological and economic considerations, was surely influenced by heightened concerns about IP in this space more generally. Eisenberg notes that the activities of the consortium members were consisting with non-binding norms known as the "Bermuda Rules," established in 1996 (Eisenberg 2000). The Bermuda rules ask companies to make human genome information public and dissuade patenting. Though the motivations behind such rules and broader public concern about biotechnology are outside the scope of this thesis, they certainly play a role in motivating disclosure and profit-driven activity in the biotechnology industry.

8.3.5 Case Summary

- **Stakeholders**
  - **Sponsors:** Group of Pharmaceutical Companies
  - **Developers:** Pharma Companies and Universities
  - **Users:** Pharma Companies
- **Objectives and Strategies**
  - **Sponsors:** Avoid patent thickets; lower input costs
  - **Developers:** Same as above
- **Technology**
  - **What was shared:** SNP Database information
  - **Modularity of Architecture:** High – database data points
  - **Standards Used:** Data-description/Measurement only
- **Intellectual Property**
  - **Architecture:** Public Domain
  - **Parts:** Public Domain
  - **Standards:** NA
- **Constraints**
  - **Value Chain:** Somewhat vertically integrated with Pharma companies processing downstream gene/drug targets; also purchase targets
  - **System Knowledge:** Low
  - **User-Need Knowledge:** High, specific diseases that could be addressed.

8.4 APIs and the Open System Environment Reference Model

8.4.1 Background

With the rise of "web 2.0" a number firms have embraced a model of development in which a core platform is released together with application programming interfaces (APIs). This is perhaps best exemplified by Google's release of its mapping API for use by third party developers.[23] Since then a number of internet firms have release APIs and developer tools including Amazon, eBay, Flicker and Yahoo.

An API is a standardized interface to access a core platform including data and services via the web. APIs define routines, protocols, object-classes, and database structures provided in libraries for use by developers to build a variety of applications.

Third party developers are encouraged to use APIs to create applications that tap into both the platform and the database associated with the platform. For example,

[23] Google maps started out with an open API and closed source code. It is now fully open source. The focus of this case in on the open API, not the source code.

by opening the APIs to it mapping program and database, Google empowered users to create "mashups" or combinations of different programs to add tremendous value to the basic platform. Housingmaps [http://www.housingmaps.com] is an example mashup of Craigslist housing posts giving users a visual representation of available housing in an area (Figure 73).

Figure 73: Screenshot from www.housingmaps.com. This map was created by searching for apartments in Boston between $1500 and $2000.

The Open-API model is extremely flexible and has been used by a wide range of companies including Google, eBay, and Amazon. A website called programmable web keeps track of most APIs (http://www.programmableweb.com). As of the time of writing this thesis, programmableweb.com listed the following

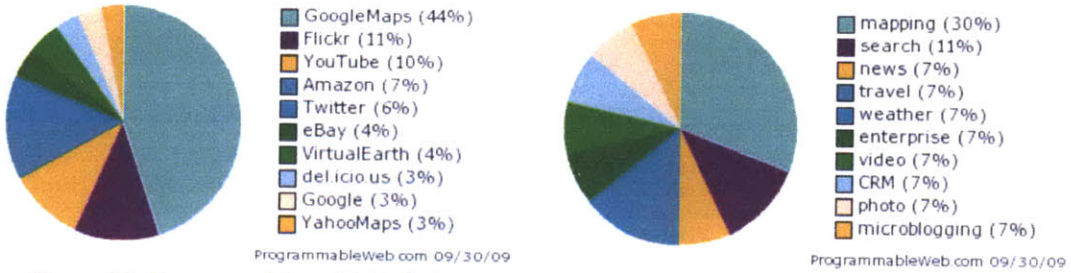


Figure 74: Decomposition of APIs (left) and total mashups (right) by company and category, respectively. Programmableweb.com has identified 2074 mashups in mapping, or about 6900 in all categories (Source: programableweb.com, accessed September 2009).

8.4.2 *Organizational and Economic Context*

The open API model is a kind of collaborative production between established internet companies with asset like data or software and third-party developers. In Google maps, and most other API-models, the "interface" occurs between a web-program and the end-user applications. In many ways, then, Google owns a complementary asset which is "upstream" of the third-party developers on the value chain.

Opening APIs lets third parties connect users with an underlying source of value. Putting the interface in the public domain and providing functional building blocks greatly reduce the barriers to innovation in a given area. As one author has noted with respect to Amazon's developer program: "With functional building blocks in hand, Amazon gives developers carte blanche to build any application they see fit. No one has to ask for permission or await approval. There's no haggling over specs or schedules." (Tapscott and Williams 2006)

8.4.3 System Architecture

The use of APIs in object-oriented programming is an established practice. Publishing APIs for third-party developers is more novel, though companies like Microsoft have been employing related strategies for years. The general goal is always the standardization of interfaces, protocols, and data-formats for automatic access to underlying resources (data and scripts). As such, they create a highly modular task environment in which third parties can create independent applications based on a common, potentially integral platforms. In many cases, firms provide tools kits for design, including functional elements of code.

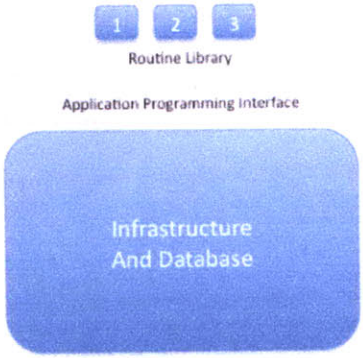


Figure 75: Schematic of the Open API strategy.

The exact nature of APIs differ depending on the kind of applications and the nature of the underlying database. However, some general structural features behind modularity alone can be identified. These are perhaps best illustrated using a set of standards known as the Open System Environment Reference Model (OSE/RM) (formerly the POSIX Open System Reference Model). OSE/RM was developed by the IEEE to better categorize and develop standard interfaces between heterogeneous

computing elements (Hungate and Gray 1995). Specifically OSE/RM describes the standards, services, protocols and data formats needed to "provide interoperability, portability, and scalability, of computerized applications across networks of heterogeneous multi-vendor hardware/software/communications platforms." (Hungate and Gray 1995)

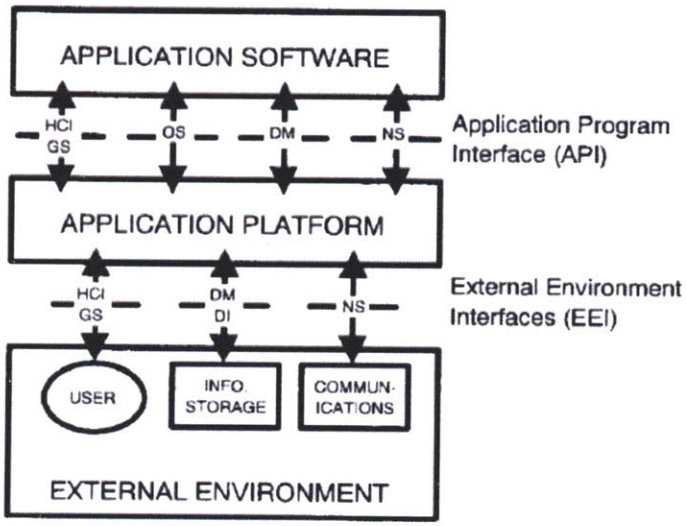


Figure 76: The Open System Reference Model (OSE/RM). Two important interfaces are defined: The API between application and platform; EEI (external environment interface) between platform and other devices and the user. (Figure from Hungate and Gray, 1995)

Figure 76 provides a schematic for OSE/RM, with the important interfaces defined. The critical interfaces are the API and EEI, with the latter including system components and the user. The model further specifies a suite of "services" which can be taken individually or in groups to define a specific implementation of an API for a program. The services fall into several distinct categories. (Hungate and Gray 1995)

1. Operating System Services
2. Human/Computer Interface Services
3. Data Management Services
4. Data Interchange Services
5. Software Engineering Services
6. Graphic Services
7. Network Services

A combination of services need for a specific domain of application, together with the interface definition, data format, and data exchange protocol is called an Application Portability Profile. (Hungate and Gray 1995)

While the specific kind of Application Portability Profiles defined for mapping versus shopping are not important to the discussion, the general approach and goals are. First, the OSE/RM formalizes the categories of interfaces needed for defining an environment in which multiple third parties can develop compatible systems. The interface, data-definition (or measurement), data-exchange (communications protocol) can together define a specific service between code or hardware elements, and suite of services can then be specified, allowing anyway to create new designs. It is likely, though would need to validated, that these basic classes are necessary, though perhaps not sufficient, to define a distributed design environment in other domains, including synthetic biology.

The basic motivation for the OSE/RM model also merits attention. The most important element associated with the model is portability. That is, the model provides a means to remove device-specific specifications from each application. By creating a standardize interface between hardware elements and platform (EEI), and platform and application (API), the OSE/RM greatly simplifies the application developers principal challenges. The fundamental problem of portability within open system development environments for synthetic biology is analyzed in chapters 5 and 6.

8.4.4 Intellectual Property

Importantly, the legal structure of the API models described vary, blending open source and proprietary licenses. For example, Google first released its APIs under a license agreement that gave developers significant freedom provided they abide by core elements of Google's privacy policies and other restrictions. Google also released functional modules to create extensions. However, the source code for these was eventually released under an open source license. Throughout, their core

mapping software remains closed source, and the database of mapping locations was licensed from a third-party.

One can speculate that Google has calculated little benefit to keeping closed "functional models" designed to encourage third-party use. Even if a company like Microsoft used them, they would be specific to the data-formats, protocols, and standards defined by the Google API. In any event, the open API strategy is clearly amenable to a mixed regime in which open and closed source co-mingle.

8.4.5 Case Summary

- **Stakeholders**
  - **Sponsors:** Internet Platform Firms – Google, Yahoo, EBay, Amazon
  - **Developers:** Third-party companies; hackers
  - **Users:** Mass Market
- **Objectives & Strategies**
  - **Sponsors:** Widen Market; Drive Traffic; Find New Outlets
  - **Developers:** Leverage infrastructure and database resource for low-cost development and profit
- **Technology**
  - **What was shared:** API and some functional elements of code
  - **Modularity of Architecture:** High
  - **Standards Used:** Interface, Data-Formats
- **Intellectual Property**
  - **Architecture:** Open/Company Licenses
  - **Parts:** Proprietary or Open Source
  - **Standards:** Open/Public Domain
- **Constraints**
  - **Value Chain:** Vertically disintegrated. However, platform firms are largely horizontally integrated.
  - **System Knowledge:** High
  - **User-Need Knowledge:** Low

8.5 Software: Opening Databases

The API strategy lowers barriers for users to create products based on access to data or a functional computer program. A simpler set of examples revolves around the basic concept of searching a database in which data-points or poorly characterized or poorly understood. At a certain level, the SNP consortium is an example of this model in that the SNP database made public for industry and academia to search. This open database model has been cited in numerous studies

of "open source" and thus merits attention as a separate class of software-based cases.

Four examples of the open database model include: Goldcorp Inc.; NASA Clickworkers; SETI@Home; Amazon Alexa. I review each only briefly here. Goldcorp is a mining company that was frustrated with their ability to locate promising veins for exploitation. They had a tremendous amount of data, more than could be realistically analyzed by their staff. In 1999 the CEO decided to publish this data and create a prize for those who could locate exploitable targets for mining (Tapscott and Williams 2006). Suggestions were made from around the world, numerous veins identified, and the company revenue grew more than ten-fold (Tapscott and Williams 2006).

NASA clickworkers [http://clickworkers.arc.nasa.gov] is an experiment by NASA Ames research center to open-source the identification of craters on Mars. Like Goldcorp, NASA has a wealth of data which would take tremendous amounts of time to fully search. Clickworkers was conceived as a way to enable the general public to carry out some of the low-skill elements of annotating the data – specifically identifying craters. Within the first six months Clickworkers attracted 85,000 users who identified over 1.9 million craters with a high level of accuracy (Benkler 2002).

SETI@Home is an early experiment in distributed data analysis. The SETI institute has the goal of identifying patters in extra-terrestrial radio signals that may indicated the presence of intelligent life on distant planets. Like NASA and Goldcorp they are mired in more data than they can realistically process. Unlike these former examples, Seti@home can automate pattern-recognition. Thus, rather than rely on users' brains, they rely on users' idle computers. A simple screen-saver was created through which data could be sent and processes by idle CPUs. SETI@home is thus really an exercise in distributed computing. As such, however, it emphasizes the similarities between massively distributed computing processes, and distributed analysis by users.

Amazon Alexa is a final example of opening a database for users to search. Through it's subsidiary search company, Alexa, Amazon has compiled a vast trove of data on the web. With the search engine languishing behind Google, Yahoo, and Microsoft, Amazon decided to open this data for any company to examine and build upon (Tapscott and Williams 2006). This is, of course, a variation on the API-theme. Tapscott and Williams identify four benefits to the strategy:

1. Access to talented developers
2. Fast innovation
3. Turns a database into a salable product
4. Less incentive for smaller rivals to build competition platforms

8.5.1 Case Summary

- **Stakeholders**
  - **Sponsors:** For profit and non-profit firms
  - **Developers:** Usually individuals or firms
  - **Users:** Sponsors
- **Objectives & Strategies**
  - **Sponsors:** Search database, find valuable elements
  - **Developers:** Depends on rewards structure: Win prizes, sell services, contribute to a good cause
- **Technology**
  - **What was shared:** Database information
  - **Modularity of Architecture:** High, from a task-structure perspective
  - **Standards Used:** Interface, Data-Format, Data-Exchange
- **Intellectual Property**
  - **Architecture:** Proprietary or public domain, depending on case
  - **Parts:** N/A
  - **Standards:** Open/Public Domain
- **Constraints**
  - **Value Chain:** Vertical disintegration possible – decoupling between database on users
  - **System Knowledge:** Low
  - **User Needs/Knowledge:** High (user is sponsor with specific objective)

9 Appendix B: Matlab Code for Chapter 6

Script 1: Meta-Script

```
clear all

%%%%%%%%%%%%%%%%%%%
% Meta Parameters %
%%%%%%%%%%%%%%%%%%%

% Typical Energy Use Per Home: 1.5 Kw/home
% Energy per gram COD = 14.7 kj/g-COD (from Logan textbook, 2008)
% Gram Hydrogen per gram COD (assuming full oxidation: 0.125 g-H2/g-COD

Ebod = 4.0705;      % WH/gram in BOD
Cbod = 0.0005;     % $/gram in savings to remove BOD
Bbod = Cbod/Ebod;  % $/WH benefit of removing BOD per Watt Hour
PE = 0.0001;     % Price of Electricity $/WH (not KWH)
Bce = .85; % Coulumbic Efficiency of the Bacterial Growth

%%%%%%%%%%%%%%%%%%%%%%%
% INPUTS Requirements %
%%%%%%%%%%%%%%%%%%%%%%%

Conc = 1000; % mg/l of input concentration
p = 1000; % Resistivity of the electrolyte (or wastewater) in OHM-CM -
typical is 1000. Could be a funct of conc

%%%%%%%%%%%%%%%%%%%%
% Design Variables %
%%%%%%%%%%%%%%%%%%%%

L = 4; % distance between electrodes in CM
Ilim = 10; % Limmiting current density in milliamps/cm^2 (This is 5
amps/m^2)
Iext = .1; % Exchange current density millamps/cm^2 (Can range from
10^-4 to 10^ -9 Amps/cm for PT)
g = .3 ; % Transfer Coefficient for the reactions related to rate-
limmitting step

Rop = 500; % Ohms - external resistor
Cost = 100; % Cost in dollars/m^2 of membrane
Cf = 1000; % Fixed cost of equipment independant of cell design

%%%%%%%%%%%%%%%%%%%%%%%%%
% Cell Voltage Paramters %
%%%%%%%%%%%%%%%%%%%%%%%%%

Ee = 1.24; % Theoretical Open Circuit Voltage without current loss
Ebact = .3; % Voltage loss due to bacteria operating potential at the
anode
Il = .24;  % Loss of voltage due to O2 and BOD crossing the membrane
En = Ee - Ebact - Il; % Real Open Circuit Voltage
cTSS = 0.00075;      % $/gram in savings for lack of TSS
```

```
OutputP = [];
OutputR = [];
OutputC = [];
MaxL = [];
MaxB = [];

for Iext = .2:.01:.5

    % Vary the length
    Ploop1 = [];
    Rloop1 = [];
    Cloop1 = [];
    Pindex = [];
    Lindex = [];
    Bindex = [];

    for L = 1:.5:5

        simpV;
        Ploop1 = [Ploop1, Pcv];
        Rloop1 = [Rloop1, BODcv];
        Cloop1 = [Cloop1, BODcv];

        tempP1 = max(Pcv);
        Pindex = [Pindex, tempP1];

        tempB1 = max(BODcv);
        Bindex = [Bindex, tempB1];

        Lindex = [Lindex, L];

    end

    tempP1 = find(Pindex == max(Pindex));
    tempB1 = find(Bindex == max(Bindex));

    temp3 = [Lindex(tempP1); Pindex(tempP1); Lindex(tempB1);
Bindex(tempB1); Iext];
    MaxL = [MaxL, temp3];

    OutputP = [OutputP, Ploop1];
    OutputR = [OutputR, Rloop1];
    OutputC = [OutputC, Cloop1];

end

% Vary the length
OutputPL = [];
OutputRL = [];
Iext = .5;

for L = 1:.1:5
        simpV;
        OutputPL = [OutputPL, Pcv];
        OutputRL = [OutputRL, BODcv];
end
```

```
% Vary theIext
OutputPE = [];
OutputRE = [];
L = 3;

for Iext = .2:.05:.5
        simpV;
        OutputPE = [OutputPE, Pcv];
        OutputRE = [OutputRE, BODcv];
end

%Iext = .5;
%for temp = 1:10
%    L = temp/2;
%    AnodeA(temp) = 100/L;
%    Cost(temp) = 100*(AnodeA(temp)^.5);
%    Cvol(temp) = Cost(temp)*(AnodeA(temp)) + Cf;
%end

scatter(OutputP, OutputR, 15, 'g');
hold on;
scatter(OutputPL, OutputRL, 50, 'r*');
scatter(OutputPE, OutputRE, 50, 'b+')
xlabel('Watts/$');
ylabel('KG-BOD/$DAY');
title('Pareto Plot for MFC Design');
hold off
```

Script 2: Reactor Model

```
i = Iext:.1:Ilim; % Creates vector of current density  mA/cm^2 of anode
%i(1) = .01;

% Calculate Volumetric Cost

AnodeA = (100/L); % This is the area of the anode in m^2 to get on m^3
of volume given the L
Cost = 100*AnodeA^.5;
Cvol = Cost*AnodeA + Cf;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Create a Polarization Curve %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Activation Losses % Using the tafel equation %

Eact = (.025/g)*log(i/Iext);
%a = (-2.3)*(0.0257/g)*log(Iext*100);
%b = 2.3*(0.0257/g);
%Eact = a + b*log(i*100) ; % Converted current to A/cm^2 instead of
mA/cm^2 I THINK that's right.
%Eact = Eact/100;

for temp = 1:length(i)
    if Eact(temp) < 0 , Eact(temp) = 0; end
end % Gets rid of positive losses
```

```
% Ohmic Losses %
Rin = L*p; % This ignores the cross-sectional area for reasons
discussed in the chapter
Eohmic = (i).*((Rin + Rop)/(AnodeA*100*100)); % Assumes that the only
contribution to Ohmic losses are internal resistance. Divides by 1000
to convert to amps. And adds operating Resistor.
                                        % The Rop is divided by area to get
the
                                        % specific resistance contribution

% Concentatration Losses %
temp1 = Ilim - i;
for temp2 = 1:length(i) % Gets rid of negative losses
    if temp1(temp2) < 0 , temp1(temp2) = 0; end
end
Econc = 0.02*log(Ilim./(temp1));

E = En - Eact - Eohmic - Econc; % Combine to create polarization curve
volts versus milliamps/cm^2
for temp = 1:length(i)
    if E(temp) < 0 , E(temp) = 0; end
end % Gets rid of positive losses

temp1 = length(find(E));
Pbod = En*i(find(E))/1000;
Pbod((temp1+1):length(E)) = Pbod(temp1);

KGbod = (Pbod*24)/Ebod; % Gram/cm^2 Day of BOD removal
BODvol = (100/L)*((100*100)/1000)*KGbod; % This is KG/m^3 - the 100*100
convers G/cm^2 t G/m^2 and the dividing by 1000 converts G to KG

%BODcv = Cvol./BODvol; % Volumetric Cost/Volumetric Rate --> $/KGDay
BODcv = BODvol/Cvol; % Volumetric Rate/Volumetric Cost --> KG/$Day
%BODcv = BODvol;


% Calculate Power Density and Electricity Value
P = (i/1000).*E; % Power density in W/cm^2
Pvol = P.*AnodeA*100*100; % Total power out per meter cubed

%Pcv = Cvol./Pvol; % Cost Density/Power Density - > $/W
Pcv = Pvol/Cvol; % Power Density/Cost Density -? W/$
%Pcv = Pvol;

%n = E/Ee; % Efficiency of the Cell
```

Script 3: Biofilm Model

```
% Biofilm %

Amp = 6.242*10^18; % Electrons per second
mAmp = Amp/1000; % milli Amps
Dh = 9.30556*10^(-05); % Diffusivity of Hydrogen Ions in E/cm^2 second
F = 96500; % Faraday's Constant


Th = 0.005; % Biofilm Thickness in CM
```

```
SPV = Th*0.001; % This is the specific volume of the biofilm in liters
per Square CM
i = 0:.01:2; % Creates vector of current density  mA/cm^2 of anode
conc = mAmp*i; % Number of electrons per second per cm^2 of biofilm
DeltaH = conc*(Th/Dh); % The delta Hydrogen through the Biofilm

K = 0.11; % Mass Transport Coefficient (or rate constant)
D = 0.88*10^(-5); % Diffusion Coefficient (less than or equal to its
value in water) P 120 has .88*10^-5 cm^2/s
bes = 8; % Number of mols electrons per mole BOD (Assuming acetate)

Imax = ((K*D)^.5)*bes*F*Bce*Conc; % This is the max current allowed by
diffusion and can be used as limmitting current A/cm^2


%Umax = 8.3; % Maximum growth rate in 1/day based on Logan Book page
114
%Kc = 400; % Half Saturation Constant - data suggests this depends on
the external resistance. Measure in mG/L
X = 30000; % dry weight of bacteria mg/l based on logan book Page 1
Yield = 1 - Bce; % Yield of bacteria per unit consumed. Bce is
coulumbic efficiency of bacteria

Cout = [];Cout;

%;

for Umax = 4:.1:10

    temp5 = [];
    for Kc = 50:10:400
        u = Umax*Conc/(Kc + Conc); % specific growth rate in 1/Day
        gRate = u*X*SPV; % growth rate in mg/cm^2 squared-day
        Consumed = (((gRate/1000)/Yield)*Bce)/136; % Amount of
substrate converted electricity each day in moles/cm^2
        Current = (Consumed*8*F/(60*60*24)); % Current in Amps/cm^2
assuming acetate is consumed which provides 8 electrons per mol
        temp5 = [temp5, Current];
    end

    Cout = [Cout; temp5];

end
```

Bibliography

Aelterman, P., K. Rabaey, et al. (2006). "Microbial Fuel Cells for Wastwater Treatment." Water Science & Technology **54** (8): 9-15.

Alexander, C. (1964). Notes on the Synthesis of Form. Cambridge, MA, Harvard University Press.

Alexander, C. (1977). A Pattern Language. New York, Oxford University Press.

Alexander, C. (1979). The Timeless Way of Building. New York, Oxford University Press.

Allen, R. C. (1983). "Collective invention." Journal of Economic Behavior & Organization **4**(1): 1-24.

Andrianantoandro, E., S. Basu, et al. (2006). "Synthetic biology: new engineering rules for an emerging discipline." Mol Syst Biol **2**.

Bactricity. (2008). "Bactricity iGem Project." from http://2008.igem.org/Team:Harvard/Shewie.

Baldwin, C. and K. B. Clark (2000). Design Rules, Vol. 1: The Power of Modularity (Hardcover). Cambridge, MA, MIT Press.

Baldwin, C. and K. B. Clark (2003). The Architecture of Cooperation: How Code Architecture Mitigates Free Riding in the Open Source Development Model, Harvard Business School: 43.

Barbir, F. (2005). PEM Fuel Cells: Theory and Practice. Burlington, MA, Elsevier Academic Press.

Barbir, F. and T. Gomez (1997). "Efficiency and economics of proton exchange membrane (PEM) fuel cells." International Journal of Hydrogen Energy **22**(10-11): 1027-1037.

Benkler, Y. (2002). "Coase's Penguin, or, Linux and "The Nature of the Firm"." The Yale Law Journal **112**(3): 369-446.

Bennetto, P. H. (1984). Microbial Fuel Cells. Life Chemistry Reports. A. M. Michelson. London, England, Harwood Academic Publishers. **2**.

Bessen, J. E. (2005). "Open Source Software: Free Provision Of Complex Public Goods." SSRN eLibrary.

Boudreau, K. J. and K. R. Lakhani (2009). "How to Manage Outside Innovation." Sloan Management Review **50**(4): 69-79.

Brandenburger, A. M. and B. J. Nalebuff (1996). Co-Opetition : A Revolution Mindset That Combines Competition and Cooperation : The Game Theory Strategy That's Changing the Game of Business. New York, Currency Doubeday Books.

Brooks, F. P. (1995). The Mythical Man-Month: Anniversary Edition. New York, NY, Addison-Wesley.

Brown, J. S. and J. I. Hagel (2005). "Innovation Blowback: Disruptive Management Practices from Asia." McKinsey Quarterly **1**.

Canton, B., A. Labno, et al. (2008). "Refinement and standardization of synthetic biological parts and devices." Nat Biotech **26**(7): 787-793.

Carlson, R. (2003). "The Pace and Proliferation of Biological Technologies." Biosecurity and Bioterrorism: Biodefense Strategy, Practice, and Science **1**(3).

Carlson, R. (2007). "Laying the foundations for a bio-economy." Systems and Synthetic Biology **1**(3): 109-117.

Chesbrough, H. (2003). Open Innovation: The New Imperative for Creating and Profiting from Technology. Cambridge, MA, Harvard Business School Press.

Chesbrough, H., W. Vanhaverbeke, et al., Eds. (2006). Open Innovation: Researching a New Paradigm. Oxford, Oxford University Press.

Chiao, M., K. B. Lam, et al. (2006). "Micromachined microbial and photosynthetic fuel cells." Journal of Micromechanics and Microengineering **16**: 2547 - 2553.

Chiao, M., L. Lin, et al. (2007). Implantable, miniaturized microbial fuel cell. US, The Regents of the University of California, Oakland CA (US).

Christensen, C. M. (1997). The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail. Boston, MA, Harvard Business School Press.

Christensen, C. M., M. Raynor, et al. (2001). "Skate to Where the Money Will Be." Harvard Business Review **79**(10): 72-81.

Clauwaert, P., P. Aelterman, et al. (2008). "Minimizing losses in bio-electrochemical systems: the road to applications." Appl. Microbiol. Biotechnol. **79**: 901-913.

Clauwaert, P. and W. Verstraete (2009). "Methanogenesis in membraneless microbial electrolysis cells." Applied Microbiology and Biotechnology **82**(5): 829-836.

Connell, C. (2000). "Open Source Projects Manage Themselves? Dream On.", 2009, from http://www.chc-3.com/pub/manage_themselves.htm.

Conway, L. (1982). The Design of VLSI Design Methods. Palo Alto, CA, Xerox PARC.

Conway, L. (1982). "The MPC Adventures." Microprocessing and Microprogramming - The Euromicro Journal **10**(4): 209 - 228.

Conway, L. (1999). Lynn Conway's Retrospective, http://ai.eecs.umich.edu/people/conway/Retrospective3.html.

Creswell, J. W. (2003). Chapter 1. Research Design: Qualitative, quantitative, and mixed methods. Thousand Oaks, CA, Sage Publications, Inc.

De Weck, O. (2004). "Multiobjective optimization: history and promise." The Third China-Japan-Korea Joint Symposium on Optimization of Structural and Mechanical Systems.

DOD (2003). Directive 5000.1 The Defense Acquisition System. E. Office.

Eisenberg, R. S. (2000). "Genomics in the public domain: strategy and policy." Nat Rev Genet **1**(1): 70-74.

Endy, D. (2005). "Foundations for engineering biology." Nature **438**(24): 449-453.

Fan, Y., E. Sharbrough, et al. (2008). "Quantification of the Internal Resistance Distribution of Microbial Fuel Cells." Environmental Science & Technology **42**(21): 8101-8107.

Feldman, M. P. (2000). Location and innovation: the new economic geography of innovation, spillovers and agglomeration. The Oxford Handbook of Economic Geography. G. L. Clark, M. P. Feldman and M. S. Gertler. Oxford, Oxford Universtity Press: 373 - 394.

Feldman, M. P. (2001). "Where Science Comes to Life: University Bioscience, Commercial Spin-offs, and Regional Economic Development." Journal of Comparative Policy Analysis **2**(3): 345-361.

Fellmeth, A. X. (2005). "The Challenge to Patent Law of Pure Chemical Protein Synthesis." Nat Biotech **23**(5): 547 - 549.

Fredrickson, J. K., M. F. Romine, et al. (2008). "Towards Environmental Systems Biology of Shewanella." Journal Name: Nature Reviews. Microbiology, 6(8):592-603; Journal Volume: 6; Journal Issue: 8: Medium: X.

Fujita, M. and K. Kageyama (1997). An open architecture for robot entertainment. Proceedings of the first international conference on Autonomous agents. Marina del Rey, California, United States, ACM.

Gamma, E., R. Helm, et al. (1995). Design Patterns: Elements of Reusable Object-Oriented Software Boston, MA, Addison-Wesley.

Gertler, M. S. and Y. M. Levitte (2003). Local nodes in global networks: The Geography of knowledge flows in biotechnology innovation. Creating, Sharing, and Transferring Knowledge. The Role of Geography, Institutions, and Organizations. Copenhagen.

Hagel, J. I. and J. S. Brown (2005). The Only Sustainable Edge. Cambridge, MA, Harvard University Press.

Hammer, M. J. and M. J. J. Hammer (2008). Water and Wastewater Technology. Upper Saddle River, NJ, Pearson Education Inc.

He, Z., Minteer, S.D. and L. T. Angenent (2005). "Electricity Generation from Artificial Wastewater Using an Upflow Microbial Fuel Cell " Environmental Science and Technology **39**(14): 5262-5267.

Henderson, R. M. and K. B. Clark (1990). "Architectural Innovation: The Reconfiguration of Existing Product Technologies and the Failure of Established Firms." Administrative Science Quarterly **35**(1): 9-30.

Henkel, J. and S. M. Maurer (2007). "The economics of synthetic biology " Molecular Systems Biology **3**.

Hiltzik, M. (1999). Dealers of Lightening: Xerox PARC and the Dawn of the Computer Age. New York, NY, Harper Collins Publishers, Inc.

Hope, E. J. (2004). Open Source Biotechnology. Faculty of Law. Canberra, Australian National University. **Doctorate of Philosophy**.

Hungate, J. I. and M. M. Gray (1995). "Application Portability Profile and Open System Environment User's Forum." Journal of the National Institute of Standards and Technology **100**(6): 699 - 709.

Izallalen, M., R. Mahadevan, et al. (2008). "Geobacter sulfurreducens strain engineered for increased rates of respiration." Metabolic Engineering **10**(5): 267-275.

Kato, A., C. Torres, et al. (2007). "Conduction-based modeling of the biofilm anode of a microbial fuel cell." Biotechnology and Bioengineering **98**(6): 1171-1182.

Keller, J. and K. Rabaey (2008). Experiences from MFC Pilot Plant Operation: How to Get the Technology Market-Ready. First International Microbial Fuel Cell Conference, State College, PA.

Kelly, J., A. Rubin, et al. (2009). "Measuring the activity of BioBrick promoters using an in vivo reference standard." Journal of Biological Engineering **3**(1): 4.

Kim, B. H. S. (1999). "Direct electrode reaction of Fe (III)-reducing bacterium, Shewanella putrefaciens." Journal of Microbiology and Biotechnology **9**: 127-131.

Knight, T. (2005). "Idempotent Vector Design for Standard Assembly of Biobricks." Unpublished.

Kordesch, K. and G. Simader (1996). Fuel Cells and Their Application. Weinheim, Germany, VCH Verlagsgesellschaft mbH.

Krishnamurthy, S. (2005). An Analysis of Open Source Business Models. Perspectives on Free and Open Source Software. J. Feller, B. Fitzgerald, S. A. Hissam and K. R. Lakhani. Cambridge, MA, MIT Press.

Kuan, J. W. (2001). "Open Source Software as Consumer Integration Into Production." SSRN eLibrary.

Kuhn, T. S. (1970). The Structure of Scientific Revolutions. Chicago, University of Chicago Press.

Lakhani, K. R. and R. G. Wolf (2003). "Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects." MIT Sloan Working Paper **4425-03**.

Langlois, R. (2002). "Modularity in Technology and Organizations." Journal of Economic Behavior and Organization **49**: 19-37.

Lerner, J. and J. Tirole (2002). "Some Simple Economics of Open Source." Journal of Industrial Economics **50**(2): 197-234.

Lerner, J. and J. Tirole (2005). "The Economics of Technology Sharing: Open Source and Beyond." The Journal of Economic Perspectives **19**(2): 99-120.

Liu, H., R. Ramnarayanan, et al. (2004). "Production of Electricity during Wastewater Treatment Using a Single Chamber Microbial Fuel Cell." Environmental Science & Technology **38**(7): 2281-2285.

Logan, B. E. (2005). "Simultaneous wastewater treatment and biological electricity generation." Wat. Sci. & Tech. **52**(1-2): 31-37.

Logan, B. E. (2008). Microbial Fuel Cells. Hoboken, New Jersey, John Wiley & Sons.

Logan, B. E. and J. M. Regan (2006). "Microbial Fuel Cells: Challenges and Applications." Environmental Science & Technology.

Logan, B. S. C. P. A. and S. M. D. E. Grot (2006). Bio-electrochemically assisted microbial reactor that generates hydrogen gas and methods of generating hydrogen gas. US\. **11180454**\.

Lovley, D. R. (2008). "The microbe electric: conversion of organic matter to electricity." Current Opinion in Biotechnology **19**(6): 564-571.

Maier, M. W. and E. Rechtin (2002). The Art of Systems Architecture, Second Edition. New York, NY, CRC Press.

Marshall, M., L. Waller, et al. (1981). For Optimal VLSI Design Efforts, Mead and Conway have fused device fabrication and systems-level architecture. Electronics.

Mead, C. (2009). Personal Interview. M. Silver.

Mead, C. and L. Conway (1980). Introduction to VLSI Systems. Reading, Addison-Welsey Publishing Company Inc.

Meyer, P. B. (2003). Episodes of Collective Invention: Working Paper. U. B. o. L. Statistics. **368**.

Miles, R. E., G. Miles, et al. (2005). Collaborative Entrepreneurship: How Communities of Networked Firms Use Continuous Innovation to Create Economic Wealth. Palo Alto, CA, Stanford University Press.

Moses, J. (2002). The Anatomy of Large Scale Systems. ESD International Symposium. Cambridge, MA, MIT.

NACWA (2005). National Survey of Municipal Wastewater Management Financing Trends. N. A. o. C. W. Agencies.

Newcomb, J. (2007). "The new age of biological engineering: Implications for the US economy." Industrial Biotechnology **3**(4): 325.

Noll, K. (2006). Microbial Fuel Cells. Fuel Cell Technology: 277-296.

Norman, D. O. and M. L. Kura (2004). Engineering Complex Systems, Mitre Corporation.

NRC (1999). Funding a Revolution: Government Support for Computing Research. C. o. I. i. C. a. Communications. Washington, D.C., National Academy Press.

Ntarlagiannis, D., E. Atekwana, et al. (2007). "Microbial Nanowires: Is the Subsurface "Hardwired"?" Geophysical Research Letters **34**: 5.

Nuvolari, A. (2004). "Collective invention during the British Industrial Revolution: the case of the Cornish pumping engine." Camb. J. Econ. **28**(3): 347-363.

Nuvolari, A. and B. Verspagen (2007). "Lean's Engine Reporter and the Development of the Cornish Engine: A Reappraisal." Transnational Necomen Society **77**: 167-189.

Oh, S. and B. E. Logan (2005). "Hydrogen and electricity production from a food processing wastewater using fermentation and microbial fuel cell technologies." Water Research **39**: 4673-4682.

ORNL. (2009). "Human Genome Project." from http://www.ornl.gov/sci/techresources/Human_Genome/faq.

Osterloh, M. and S. Rota (2007). "Open source software development--Just another case of collective invention?" Research Policy **36**(2): 157-171.

Oye, K. and R. Wellhausen (2009). The Intellectual Commons and Property in Synthetic Biology. Forthcomming In: Synthetic Biology: The Technoscience and its Societal Consequences M. Schmidt, Springer Academic Publishing.

Peccoud, J., M. F. Blauvelt, et al. (2008). "Targeted Development of Registries of Biological Parts." PLoS ONE **3**(7): e2671.

Picioreanu, C., I. M. Head, et al. (2007). "A computational model for biofilm-based microbial fuel cells." Water Research **41**(13): 2921-2940.

Pisano, G. P. and R. Verganti (2008). "Which Kind of Collaboration is Right for You?" Harvard Business Review.

Potter, M. C. (1911). "Electrical effects accompanying the decomposition of organic compounds." Royal Society (Formerly Proceedings of the Royal Society) **84**: 260-276.

Powell, W. W., K. W. Koput, et al. (1996). "Interorganizational Collaboration and the Locus of Innovation: Networks of Learning in Biotechnology." Administrative Science Quarterly **41**(1): 116-145.

Rai, A. and J. Boyle (2007). "Synthetic Biology: Caught between Property Rights, the Public Domain, and the Commons." PLOS Biology **5**(3): 0389-0393.

Ranjay, G. (1998). "Alliances and networks." Strategic Management Journal **19**(4): 293-317.

Raymond, E. S. (1999). The Cathedral & the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary Sebastopol, CA, O'Reilly & Associates Inc.

Reimers, C. E., L. M. Tender, et al. (2000). "Harvesting Energy from the Marine Sediment, Water Interface." Environmental Science & Technology **35**(1): 192-195.

Resnick, P., N. Iacovou, et al. (1994). GroupLens: an open architecture for collaborative filtering of netnews. Proceedings of the 1994 ACM conference

on Computer supported cooperative work. Chapel Hill, North Carolina, United States, ACM.

Ringeisen, B. R., E. Henderson, et al. (2006). "High Power Density from a Miniature Microbial Fuel Cell Using Shewanella oneidensis DSP10." Environmental Science & Technology **40**(8): 2629-2634.

Rossi, M. (2004). "Decoding the 'Free/Open Source (F/OSS) Software Puzzle'a survey of theoretical
and empirical contributions." http://opensource.mit.edu/papers/rossi.pdf.

Rozendal, R. A., E. Leone, et al. (2009). "Efficient hydrogen peroxide generation from organic matter in a bioelectrochemical system." Electrochemistry Communications **11**(9): 1752-1755.

Sanchez, R. and J. T. Mahoney (1996). "Modularity, Flexibility, and Knowledge Management in Product and Organization Design." Strategic Management Journal **17**: 63-76.

Schofield, S. and P. Wright (1998). "Open Architecture Controllers for Machine Tools, Part 1: Design Principles." Journal of Manufacturing Science and Engineering **120**(2): 417-424.

Shapiro, C. and H. R. Varian (1999). Information Rules: A Strategic Guide to the Network Economy. Cambridge, MA, Harvard Business School Press.

Shirky, C. (2008). Here Comes Everybody. New York, NY, Penguin Books.

Simon, H. A. (1962). "The Architecture of Complexity." Proceedings of the American Philosophical Society **106**(6): 467-482.

Tapscott, D. and A. D. Williams (2006). Wikinomics: how mass collaboration changes everything. New York, NY, Penguine Group.

Thomas, J. M. (1995). A Nation of Steel: The Making of Modern America, 1865-1925. Baltimore, Johns Hopkins University Press.

Torres, C. I., A. K. Marcus, et al. (2008). "Proton transport inside the biofilm limits electrical current generation by anode-respiring bacteria." Biotechnology and Bioengineering **100**(5): 872-881.

Torvalds, L. (1999). The Linux Edge. Open Sources: Voice from the Revolution. C. DiBona, S. Ockman and M. Stone. Sebastopol, CA, O'Reilly Associates.

USDA (2008). U.S. Biobased Products Market Potential and Projections Through 2025. U. S. D. o. A. O. o. E. a. Policy. Washington, D.C.

Virdis, B., K. Rabaey, et al. (2008). "Microbial fuel cells for simultaneous carbon and nitrogen removal." Water Research **42**(12): 3013-3024.

Von Hippel, E. (1990). "Task partitioning: An innovation process variable." Research Policy **19**(5): 407-418.

Von Hippel, E. (2005). Democratizing Innovation. Cambridge, MA, MIT Press.

Von Hippel, E. and K. Georg von (2003). "Open Source Software and the "Private-Collective" Innovation Model: Issues for Organization Science." Organization Science **14**(2): 209-223.

Weber, S. (2004). The Success of Open Source. Cambridge, MA, Harvard University Press.

Weijan Shan, G. W. B. K. (1994). "Interfirm cooperation and startup innovation in the biotechnology industry." Strategic Management Journal **15**(5): 387-394.

Whitten, D. O., B. E. Whitten, et al. (2005). The birth of big business in the United States, 1860-1914. New York, NY, Praeger Publishers.

Wilkinson, S. (2000). ""Gastrobots"—Benefits and Challenges of Microbial Fuel Cells in FoodPowered Robot Applications." Autonomous Robots **9**(2): 99-111.

Young, R. (1999). Giving it Away: How Red Hat Software Stumbled Across a New Economic Model and Helped Improve an Industry. Open Sources: Voice from the Revolution. C. DiBona, S. Ockman and M. Stone. Sebastopol, CA, O'Reilly Associates.