Presented at the 2007 ISPA/SCEA Joint Annual International Conference and Workshop - www.iceaaonline.com

Pioneers of Parametrics

Ricardo Valerdi
MIT
Cambridge, MA 02139
rvalerdi@mit.edu

This paper provides a historical account of the development of the field of parametrics through information obtained during interviews of twelve pioneers of the field. Cost model developers, users, and practitioners were interviewed with the intent to capture their views on the impact between cost estimation research and practice. The individuals interviewed represent a diverse range of perspectives including academia, government, and industry. Each perspective sheds light on the areas in which the field of parametrics has had an impact and which synergies have been influential in the development of the field. The implications of the findings are discussed in light of the future challenges for the field of parametrics.

Introduction

Professional societies dedicated to the promotion and enhancement of parametrics have been around for almost 40 years. Initially, through the creation of the National Estimating Society in 1968 and subsequently the Institute of Cost Analysis which eventually merged to become the Society of Cost Estimating and Analysis. Shortly thereafter, the International Society of Parametric Analysts and closely-related Society for Cost Analysis and Forecasting began to have a critical role in advancing the field of parametrics.

The origins of parametrics predate these societies but much of the history behind the evolution of the field has been anecdotal. It is possible to trace the history of the field through conference proceedings, textbooks, and journal publications. From a historian's perspective these primary sources are ideal for rebuilding accounts of events and people. However, we are in an even better situation today because most of the individuals that were involved in the development of the field are still alive. Many of them are retired or approaching retirement which makes it even more necessary to obtain first-hand accounts of the development of the field from its genesis. These individuals, the pioneers of the field of parametrics, are the best source of information about the development of the most important innovations that have shaped the field of parametrics.

Three people in particular were not able to be part of this study. One is Bob Park who was involved with the development of the PRICE model. He is believed to be retired and living somewhere in New England but several attempts to reach him have proved unsuccessful. Another is Alan Albrecht who developed function points while at IBM. The last is Dick Stutzke who was the recipient of the ISPA Frieman award in 2006 and the author of *Estimating Software-Intensive Systems* (Addison Wesley 2005). He passed away in 2006 of cancer. Despite these missing links, pioneers across industry, government, and academia provided their perspectives on the field of parametrics which are presented in this paper.

Method

Following the premise that the history of parametrics can be best told by the pioneers of the field, individuals were seeked out that had in some way been involved with the development of the field through the creation, evaluation, or enhancement of new techniques or models that contributed to the way parametrics is done today. After initial interviews, it was clear that the development of the field did not rest solely on the shoulders of model developers. It became apparent that two additional groups, users and evaluators, played an equally important role in the improvement of the models and the refinement of techniques that have influenced the field of parametrics. Each interview with a pioneer also helped identify other individuals that played a significant role in the development of the field. Eventually, the same names surfaced which served as a confirmation that the right individuals were included.

Of the twelve pioneers interviewed, most of them fell into more than one group even though they identified themselves as having a single primary group. The names of the twelve pioneers and their affiliations are listed in Table 1.

Table 1. Pioneers of Parametrics

Name (Affiliation)	User	Evaluator	Developer
Vic Basili (University of Maryland)		X	X
Barry Boehm (TRW/USC)			X
Stephen Book (MCR Federal)		X	
Dan Galorath (Galorath, Inc.)			X
Jairus Hihn (NASA JPL)	X	X	
Randy Jensen (Hughes/Jensen Consulting/US Air Force)			X
Capers Jones (Software Productivity Research)			X
Larry Putnam (Quantitative Software Measurement, Inc.)			X
Don Reifer (RCI Consulting)			X
Dieter Rombach (Fraunhofer Institute/TU Kaiserslautern)		X	
Walker Royce (TRW/Rational/IBM)	X		X
Marilee Wheaton (TRW/The Aerospace Corporation)	X		

In-person interviews were conducted with each pioneer guided by the questions listed in Appendix A. The questions provided a general framework for the interview, but each pioneer steered the interview to a unique trajectory. The preceding sections provide general themes that emerged from the interviews which shed light on the development of the field. Rather than attributing each pioneer's viewpoint of the field, it was determined that their inputs would be better presented in four cross-cutting themes: (1) major achievements in parametrics, (2) synergies between industry/government/academia, (3) impact of research on practice and practice on research, and (4) challenges for the future.

Major Achievements in Parametrics

The major achievements in the field of parametrics are organized into two main eras: early achievements and intermediate achievements. Each is marked by twenty year increments to help organize the significant events described by the pioneers. The era of early achievements includes the search for good model forms, the development of model evaluation criteria, and the emergence of a model marketplace and community of interest. The era of intermediate achievements includes mainstream refinements and proliferation of software development styles. The current era is discussed in the final section on challenges for the future.

Early achievements: 1965-1985

The earliest achievements in software resource estimation are tied to specific models developed, calibrated, and published as early as 1965, the most prominent of which are shown in Table 2. This constellation of models mark the first generation of parametric models and played an important role in establishing the field.

Table 2. Early parametric models

Model	Year
System Development Corporation[1]	1965
Martin Marietta[2] PRICE H	1973
TRW[3] Wolverton	1974
Putnam	1976
Doty	1977
RCA PRICE S	1977
IBM-FSD	1977
Boeing-Black	1977
IBM Function points	1979
SLIM	1980
Bailey-Basili Meta-Model	1981
SoftCost, -R	1981
COCOMO	1981
Jensen/SEER-SEM	1983
ESTIMACS	1983
SPQR/Checkpoint	1985

[1] System Development Corporation was a 1957 spinoff of the RAND Corporation and was sold to Burroughs Corporation, Unisys, Loral, and eventually Lockheed Martin

[2] Martin Marietta merged with Lockheed Corporation in 1995 to become Lockheed Martin

[3] Originally known as Thompson Ramo Wooldridge, TRW was acquired by Northrop Grumman in 2002

Early users of these models began to introduce necessary improvements. The first involved the development of criteria for determining good model forms.

The Search for Good Model Forms

The most critical issue in the early stages of parametrics was to find the right forms for estimating software project effort and schedule. The experiences in analyzing the SDC database convinced people that a purely linear additive model did not work well (Boehm 1981). It was clear that the behavioral phenomenology of software development was not consistent with effort estimators combining such factors as size and complexity in linear additive forms. For a while, the best relationships people could find involved estimating effort as a linear function of size, modified by a complexity multiplier. The initial complexity multiplier came from the nonlinear distribution of programming rates in the 169-project SDC sample shown in Figure 1. For example, if one's project involved developing 10,000 object instructions of software that was considered to be more complex than 80% of the projects in the SDC sample, one would determine that its programming rate would be roughly 7 person-months per thousand object instructions. Then the estimated project effort would be 7*10=70 person-months.

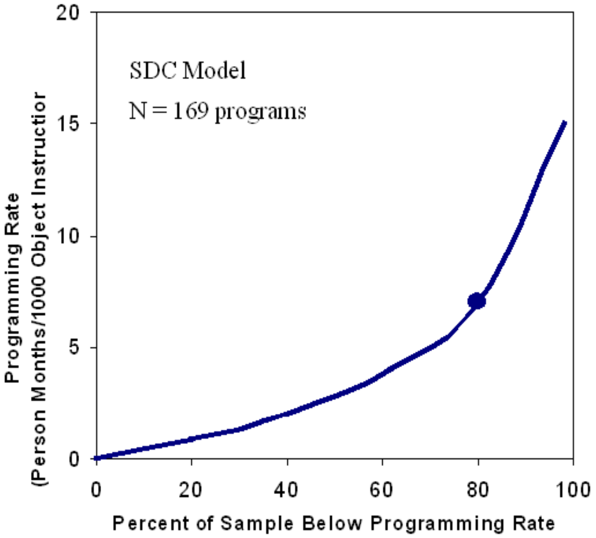


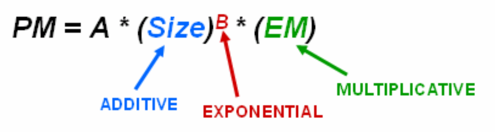
Figure 1. SDC Model Example

Most of the successful early effort estimation models employed variants of this approach. The TRW Wolverton model (Wolverton 1974), the Boeing Black model, and early versions of the RCA PRICE S model employed different programming rate curves for different classes of software (scientific vs. business vs. embedded real-time; familiar vs. unfamiliar; and/or for different software life-cycle phases).

By the late 1970's, the software community was finding that simple complexity ratings were not adequate for many software situations that produced different programming rates. Some organizations found that their programming rates were more productive rather than less productive for higher-complexity software, as they assigned their best people to the most complex projects. Most importantly, though, the complexity rating was purely subjective. There was no objective way of determining if a project was at the 60% or 80% level, but going from 80% to 60% in Figure 1 will reduce the estimated effort by roughly a factor of 2. Some organizations were also finding that their software projects exhibited economies or diseconomies of scale, involving estimation relations with exponential functions of size.

Quite a few estimation models were developed in the late 1970's. Multiple combinations of multiplicative cost drivers were employed in the Doty (Doty 1977), IBM, Walston-Felix, and intermediate versions of the PRICE S model (Walston & Felix 1977). Bailey and Basili experimented with an additive combination of productivity multipliers and an exponential scale factor (Bailey & Basili 1981). The Putnam SLIM model developed exponential relationships linking size, productivity, and schedule (Putnam 1978). Alternative sizing methods such as function points (Albrecth & Gaffney 1983) were being developed to better support early size estimation.

Positive and negative experiences with these and other models led to a set of criteria for developing additive, exponential, multiplicative, and asymptotic model factors. This underlying logic provided the general form for the Constructive Cost Model (COCOMO) in 1981.



Where:

PM = Person Months
A = calibration factor
Size = measure(s) of functional size of a software module that has an additive effect on software development effort
B = scale factor(s) that have an exponential effect on software development effort
EM = effort multipliers that influence software development effort multiplicatively.

The general rationale for whether a factor is additive, exponential, or multiplicative comes from the following criteria:

1. A factor is additive if it has a local effect on the included entity. For example, adding another source instruction, function point entity, module, interface, operational scenario, or algorithm to a system has mostly local additive effects.

2. A factor is multiplicative or exponential if it has a global effect across the overall system. For example, adding another level of service requirement, development site, or incompatible customer has mostly global multiplicative or exponential effects. Consider the effect of the factor on the effort associated with the product being developed. If the size of the product is doubled and the proportional effect of that factor is also doubled, then it is a multiplicative factor.

3. If the effect of the factor is more influential for larger-size projects than for smaller-size projects, often because of the amount of rework due to architecture and risk resolution, team compatibility, or readiness for system-of-systems integration, then it is treated as an exponential factor.

4. The effects of asymptotic cost driver forms generally interact multiplicatively with other cost factors. Their effects on cost tend to increase unboundedly as they reach constraint boundaries. Examples are reaching limits on available computer execution cycles or main memory capacity, or on achievable schedule compression. Such factors can be calibrated as a multiplicative coefficient times the shape of the asymptotic curve.

These rules, which have also been applied to other models in the COCOMO suite, require that the assumptions made about the Cost Estimating Relationships (CER) be validated by historical projects. A crucial part of developing these models is finding representative data that can be used to calibrate the size, multiplier, and exponential factors contained in the models. As with other parametrics models, industry data and usage experience allowed model developers to validate the model and experiment with data sets to elaborate the concepts presented as well as investigate new ones.

Development of Model Evaluation Criteria

Models are frequently evaluated for the goodness of their ability to estimate software development. Model accuracy, however, was not the only measure of quality for parametric models. As more models came into fruition and their complexity increased, developers, users, and evaluators identified criteria that was most helpful in evaluating the utility of a parametric model for practical estimation purposes:

1. *Definition*. Has the model clearly defined the costs it is estimating, and the costs it is excluding? One model's answer to this question was "What would you like it to include?" -- not a strong confidence-booster.
2. *Fidelity*. Are the estimates close to the actual costs expended on the projects? An important follow-up question is the next question on Scope.
3. *Scope*. Does the model cover the class of projects whose costs you need to estimate? Parametric models are generally not as accurate for very small and very large applications. Function points are a better match to business applications and early estimation of GUI-based applications than to algorithm-intensive scientific applications.
4. *Objectivity*. Does the model avoid allocating most of the software cost variance to poorly calibrated subjective factors (such as complexity)? That is, is it harder to jigger the model to obtain any results you want?
5. *Constructiveness*. Can a user tell why the model gives the estimates it does? Does it help the user understand the job to be done? Neural net models whose internals bear no relation to software phenomenology are a counterexample. Proprietary models were initially reluctant to discuss their internals, but have become increasingly communicative.
6. *Detail*. Does the model easily accommodate the estimation of a system consisting of a number of subsystems and units? Does it give (accurate) phase and activity breakdowns? A limiting factor here is that the greater the model detail, the less

data there is to support its calibration. Newer process models such as overlapping incremental development make the data less precise.

7. *Stability*. Do small differences in inputs produce small differences in output cost estimates? Some models, such as the Doty model, exhibited factor-of-2 discontinuities at model boundaries due to the binary nature of their inputs.
8. *Ease of Use*. Are the model inputs and options easy to understand and specify? Some related criteria are tailorability and composability for ease of calibration; addition of new parameters; composition with sizing models, risk analyzers, or generators of proposals and project plans; or model simplification for ease of early estimation, as discussed next.
9. *Prospectiveness*. Does the model avoid the use of information which will not be known until the project is complete? The greatest difficulty here is with source lines of code (SLOC) as a model's size parameter. A number of higher level quantities have been pursued, such as number of requirements, use cases, web objects, function points, feature points, object points, and application points. A major difficulty with such parameters is that their number increases as the product is better defined. Coming up with easy-to-understand definitions of the right level of detail is difficult.
10. *Parsimony*. Does the model avoid the use of highly redundant factors, or factors which make no appreciable contribution to the results? The Walston-Felix model had four separate factors which were highly correlated for IBM projects: use of top-down development, structured programming, structured walkthroughs, and chief programmer teams. This can cause both multiple-counting of correlated effects or difficulties in performing regression analyses.

For the most part, the significance of each of these criteria is reasonably self-evident. The criteria have also proven to be very helpful in the development and evaluation of commercial parametric models and related communities of interest.

Emergence of a Model Marketplace and Community of Interest

The early 1980's marked an important stage in the development of a parametrics community of interest, including conferences, journals, and books; the most influential of which are listed in Appendix B. These helped in socializing the issues above, and the emergence of several estimation models that passed both usage tests and tests of market viability. These included the refinement of earlier models such as PRICE S and SLIM, and the development of early-1980's models such as SPQR/Checkpoint, ESTIMACS (Rubin 1983), Jensen/SEER, Softcost-R, and COCOMO and its commercial implementations such as PCOC, GECOMO, COSTAR, and Before You Leap. These models were highly effective for the largely waterfall-model, build-from-scratch software projects of the 1980's and defined the early achievements of the field of parametrics. But as systems became more complex and new techniques came to light, the models began to encounter new classes of challenges, as discussed in the next section on intermediate achievements.

Intermediate Achievements: 1985-2005

Mainstream Refinements: 1985-1995

The 1985-1995 time period primarily involved proprietors of the leading cost models addressing problem situations brought up by users in the context of their existing mainstream capabilities. Good examples are the risk analyzers, either based on Monte Carlo generation of estimate probability curves, or based on agent-based analysis of risky combinations of cost driver ratings. Another good example is the breakdown of overall cost and schedule estimates by phase, activity, or increment.

The most significant extensions during this period were in the area of software sizing. Accurate early estimation of SLOC was a major challenge as new programming languages emerged (Putnam 1978). Some comparison-oriented methods involving paired size comparisons, ranking methods, and degree-of-difference comparisons were developed. They have been helpful, but their performance is spotty and expert-dependent. During 1985-1995, the Function Point community made a major step forward in defining uniform counting rules for its key size elements of inputs, outputs, queries, internal files, and external interfaces, along with associated training and certification capabilities. This made Function Points a good match for business applications, which tend to have simple internal business logic, but less good for scientific and real-time control applications with more complex internals (Jones 1991).

Function point extensions such as feature points, COSMIC function points, and 3D function points have been developed for these, but their definitions and counting rules have not converged as well as those of the initial function points quantities. Other higher-level early sizing metrics have been developed, but their counting rules and granularity standards have been more difficult to standardize than those of function points. Within individual organizations, some of the higher-level early sizing metrics have been made to work, but the challenge of developing a general early software sizing metric remains high.

Proliferation of Software Development Styles: 1995-2005

The improvement of existing parametric models was based primarily on the realization that the underlying assumptions of the existing models were based on sequential waterfall-model development and software reuse with linear savings were becoming obsolete. The projection of future hardware components also shaped the development of several new parametric models. This involved developing new cost estimating relationships for hardware; developing a more realistic nonlinear reuse models; adding exponential scale factors for such scalability controllables as process maturity and architecture/risk resolution; adding new cost drivers for such phenomena as development for reuse, distributed development, and personnel continuity; dropping obsolete cost drivers such as turnaround time, and enabling the use of alternative early sizing methods in software (e.g., function points) and hardware (i.e., weight, printed circuit boards, communication channels).

Each of these changes was supported by valuable research results, and by experiences in trying to tailor parametric models to new situations. Nonlinear effects of software reuse were based on research at the Software Engineering Laboratory at the NASA/Goddard Space Flight Center (Selby 1988), maintenance effort distributions (Perikh & Zvegintsov 1993), and nonlinear software integration effects (Gerlich & Denskat 1994). Anchor point milestones and their phase distributions were supported by work at Rational (Royce 1998), AT&T, and spiral model usage at TRW. Process maturity characterization was supported by collaboration with the Software Engineering Institute at CMU and its associated definitions and data on productivity effects (Hayes & Zubrow 1995). Estimation of the relative cost of writing for reuse and modeling project diseconomies of scale were supported by experiences provided by the user community.

This time period also saw an emergence of complementary methods, although regression-based approaches continued to dominate. From the 1980s to the present, half of the research being published in academic journals involved regression-based models but analogy and expert judgment models have increased over time (Jorgensen & Shepperd 2007) as shown Table 3. Two concerns arise from the spectrum of topics included in the 304 papers across 76 journals. One is that only 13 researchers had more than five publications. The other is that the number of theory-based papers in cost estimation has steadily decreased over the last twenty years. This shows that the first wave of ideas is behind us, while it is expected that a new wave will arise to address future challenges, the small number of researchers making consistent contributions is alarming given the increasing number of methods and processes being associated with today's systems.

Table 3. Estimation Approaches
(adapted from Jorgensen & Shepperd 2007)

Estimation Approach	**-1989**	**1990-1999**	**2000-2004**	**Total**
Regression	21 (51%)	76 (74%)	51 (51%)	148 (49%)
Analogy	1 (2%)	15 (9%)	15 (15%)	31 (10%)
Expert Judgment	3 (7%)	22 (13%)	21 (21%)	46 (15%)
Work Break-down	3 (7%)	5 (3%)	4 (4%)	12 (4%)
Function Point	7 (17%)	47 (29%)	14 (14%)	68 (22%)
Classification	0 (0%)	5 (3%)	9 (9%)	14 (5%)
Simulation	2 (5%)	4 (2%)	4 (4%)	10 (3%)
Neural Network	0 (0%)	11 (7%)	11 (11%)	22 (7%)
Theory	20 (49%)	14 (9%)	5 (5%)	39 (13%)
Bayesian	0 (0%)	1 (1%)	6 (6%)	7 (2%)
Hybrid	0 (0%)	3 (2%)	2 (2%)	5 (2%)
Other	2 (5%)	7 (4%)	16 (16%)	25 (8%)

Alternative Model Forms

Explorations have been made into separate alternative software resource estimation model forms. Analogy or case-based estimation (Mukhopadhyay, Vincinanza & Prietula 1992; Shepperd & Schofield 1997) uses metadata about a project to be estimated (size, type, process, domain, etc.) to base an estimate of its required resources on the resources required for the most similar projects in a large database such as the ISBSG database of 3000 software projects (ISBSG 2005). In one study (Ruhe, Jeffery & Wieczorek 2003), analogy-based estimation performed better than alternative estimation methods in 60% of the cases, but worst in 30% of the cases, indicating some promise but need of further refinement.

Neural net models use layouts of simulated neurons and training algorithms to adjust neuron connection parameters to learn the best fit between input parameters and values to be estimated. In some situations, accuracies of ±10% have been reported (Wittig 1995), but in many cases, estimation of projects outside the training set has been much less accurate. Neural net models do not provide constructive insights on the software job to be done, thus violating the *constructiveness* criterion. Other machine learning techniques have recently been used to successfully determine reduced-parameter versions of parametric cost models (Menzies, Chen, Port, et al 2005). Systems dynamics models integrate systems of differential equations to determine the flow of effort, defects, or other quantities through a process as a function of time. They are very good for understanding the effects of dynamic relations among software development subprocesses, such as the conditions under which Brooks' Law – adding more people to a late software project will make it later – holds true (Madachy 2007). Pioneering work in this area has been done in (Abdel-Hamid & Madnick 1991) for general software project relationships, and in (Madachy 1996) for interactions among effort, schedule, and effect density, in performing software inspections. Each of these model forms provides complementary perspectives to those of parametric models. Challenges for these approaches, parametrics in particular, include finding better ways to integrate their contributions.

In addition to the work by model developers in academia, proprietary model developers improved the accuracy of their models through join research collaborations. Counterpart commercial software cost model companies, such as CostXpert, Galorath, PRICE Systems, and Softstar Systems, participated as Affiliates of the USC Center for Software Engineering[4], where many new model extensions were developed to support their commercial offerings. This enabled developers and evaluators to share expertise while offering users a range of solution approaches. This synergy has been the catalyst for innovation in the field of parametrics during the intermediate achievements era and is expected to be the most significant achievement in the next era as rigor and applicability improve. This collaborative phenomenon is discussed in detail in the next section.

[4] In 2006, the CSE became the Center for Systems & Software Engineering

Synergies between Industry, Government, and Academia

The most common thread communicated by the pioneers during the interviews was the importance of the synergy between industry, government, and academia. Although each pioneer identified themselves with one community, some had experience in more than one. Naturally, most users were employed in industry while most developers were independent consultants or academics. On the other hand, evaluators existed in all three communities. This section explores the emergence of these roles and the impact they had on the evolution of the field of parametrics.

Emergence of Industry Roles

The long time concerns for industry users have been on planning, proposals, overrun avoidance, and program executability. This has motivated their interest in using parametric models to better understand the total cost of ownership and anticipate new trends. The evolution of industry roles in the development of parametric models was initially marked by the technological innovations in large projects in the 1970's at IBM, AT&T, Hughes Aircraft[5], and TRW. These organizations funded much of the early research in parametrics. In the 1980's the emphasis shifted to business software and techniques such as function points and estimation by analogy came into favor. In the 1990's many telecommunication, e-commerce, and services companies attracted attention to “product time to market” which spawned new approaches. In the 2000's rapid change techniques such as agile methods sparked interest in dynamic estimation approaches. Commercial cost model companies previously mentioned played a special role in addressing cross-company and cross-business unit data opportunities.

For some, the most significant contribution from industry to the advancement of parametrics was the the data from historical projects. It was the most relevant and provided a seedbed for open models such as function points at IBM and the Constructive Cost Model at TRW. In some cases the data was limited in scope and proprietary issues ocassionally slowed progress but over time industry became more open to sharing best practices as professional societies for parametrics provided a neutral forum for knowledge sharing.

Emergence of Government Roles

The role of government in the development of parametrics is rooted in the planning of large acquisitions, source selection, and avoiding overruns on projects. Traditionally, government has been the central source for funding much of the research and development in parametrics. Sponsorship of new model research and evaluations was channeled to academia, industry (through small business grants), and the foundation of nonprofits.

The evolution of government roles in parametrics began in the 1960's with the Air Force funded study by the System Development Corporation and subsequent software initiatives funded by NASA Software Engineering Lab and the US Army in the 1970's. At the same time, the formation of the Space Systems Cost Analysis Group solidified the emphasis of parametrics on space software initiatives. In the 1980's federal governments established cost analysis offices in the DoD, NASA, FAA, and UK Ministry of Defense.

[5] Hughes Aircraft was acquired by General Motors and was partially sold to Raytheon in 1997 and The Boeing Company in 2000

Government-sponsored software productivity initiatives also emerged in the European community and in specific countries such as Norway, UK, Japan, China, and Australia. As maturity models came into fashion in the 1990's the usage of parametric models significantly increased and stimulated their improvement.

Emergence of Academia Roles

The role of academia in the development of parametrics has been defined by interests in the scientific understanding of cost phenomena. This results in a focus on model exploration, future model extensions, and model evaluation. In the 1970's simple models were created by researchers at Purdue University, the University of Maryland, and the University of North Carolina which served as stepping stones for future work in parametric cost models. In the 1980's work from the University of California Los Angeles, the University of Southern California, and the Wang Institute of Graduate Studies[6] influenced many industry model extensions such as COCOMO and function points. In the 1990's new models were developed to address new system integration issues such as Commercial-Off-The-Shelf, techniques such as Rapid Application Development, and domains such as Systems Engineering.

Academia faces several key challenges as a member of the parametrics community. Access to data will continue to be a barrier due to its sensitive nature in industry and government. The knowing-doing gap between academics and practitioners will also affect context understanding of model needs. The focus on knowledge creation over knowledge deployment in academia will also affect the sustainment of models developed at universities due to the focus on publication.

An illustration of the early synergies between industry, government, and academia for the creation and refinement of parametric models is provided in Figure 2.

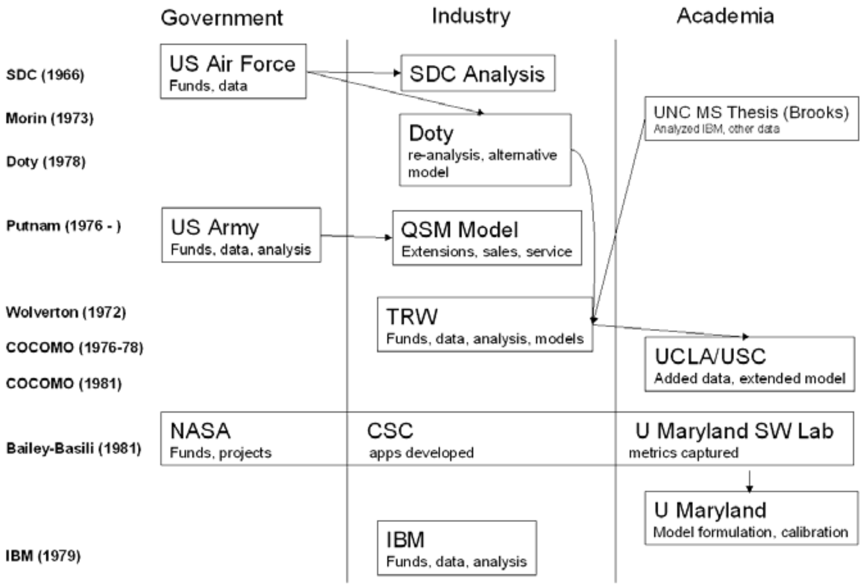


Figure 2. Government/Industry/Academia Synergies in Parametrics

[6] The Wang Institute of Graduate Studies existed from 1980 to 1988 and merged with Boston University

Parametric models could not reach a satisfactory level of maturity without the collaboration of multiple entities. What is not apparent is the increasingly important role that professional societies played in the development of the field.

Key roles of professional societies

Professional societies such as ISPA, SCEA, and its predecessors serve an important role in the knowledge creation process. They help in cross-leveling knowledge across communities of practitioners and researchers by providing a neutral forum for collaboration between users, evaluators, and developers. Often done through publications and meetings, they are an enabler for sharing tacit knowledge and integrating best practices into training modules or certification programs. The most influential conferences and journals in parametrics are listed in Appendix B.

Key role of databases

Databases of historical projects also play an important role in the inception and refinement of parametric models. Model developers cannot do good research without sound, comparable data. Unfortunately, most databases are proprietary and academic access usually requires data protection and non-disclosure agreements. In addition to accessibility challenges, some databases are hard to extend to other domains, lack the necessary detail for extensive analysis, suffer from data quality issues, or are limited in scope due to the information included. Despite these issues, parametric models have depended on databases in the past and will continue to in the future as information about recently completed projects helps in the verification and validation of parametric models.

Impact of Research on Practice and Practice on Research

Up to this point, the discussion has been around the roles of different stakeholders on the advancement of the field of parametrics. A complementary discussion is on the impact that research has had on the practice of parametrics and vice versa. This perspective provides additional insight into the role that pioneers played in the evolution of the field. Practitioners identified the following areas of practice that were influenced by research:

- *Basis of project stakeholder negotiation and expectations management*. Ability to avoid overcommitment to infeasible budgets and schedules; realistic risk mitigation for potential slips.
- *Reduced resistance to measurement*. Along with the acceptance of parametrics in the technical community, the necessary coexistence of a measurement culture.
- *Increasingly sophisticated review boards*. Organizations demanded increasingly more detailed and sophisticated cost and schedule estimates at early stages of the program.
- *Improved project performance*. Phase and activity estimates provide a framework for better progress monitoring and control of projects.

- *Framework for process improvement*. Enablement of improved planning realism, monitoring and control, model improvement, and productivity/cycle time/quality/business value improvement. Ultimately, CMMI Level 5 attributes were influenced by the practices contained in parametric models.
- *Contributions to communities of interest*. Besides the core parametrics community, these include the communities concerned with empirical methods, metrics, economics-driven or value-based software engineering, systems architecting, software processes, and project management.

The impact of practice had an equally important impact on research which provides slightly different insights into the development of the field.

- *Multiple modes of estimation.* Different project situations called for diverse estimation approaches such as analogy, expert judgment, and hybrid methods which complemented parametric approaches.
- *Bayesian approximation*. The need to combine expert judgment with historical project data motivated the development of mathematical approaches and related research areas.
- *Basis of project planning and control and impact on processes*. Realities of projects motivated the need for the alignment of project milestones and activity allocations to enable the control of complex concurrent engineering processes. Schedule/cost/quality as independent variable processes enabled meeting targets by prioritizing and adding or dropping marginal-priority features.
- *Flexibility*. The need to accommodate over 650 programming languages, compatibility with standards, new techniques (i.e., rapid application development), and local calibration factors required models to be more adaptable to change.
- *Fit to reality*. Regardless of the mathematical elegance of models, their constructiveness and inclusion of management factors needed to pass the common sense test. The aforementioned model evaluation criteria are a good example.

In the eyes of the pioneers, the complementary impact between research and practice helped advance the field in unprecedented ways. Despite the advancements over the last four decades, many challenges remain for the evolution and sustainment of the field of parametrics.

Challenges for the Future

The pioneers expect that, as the technical environment continues to change, future challenges will introduce new opportunities for improved parametric methods and tools. Such challenges are organized into two categories: evolution and sustainment. The evolutionary challenges are:

- *Integration of software and systems engineering estimation*. Challenges include compatible sizing parameters, schedule estimation, and compatible outputs.
- *Sizing for new product forms*. These include requirements or architectural specifications, stories, and component-based development sizing.
- *Exploration of new model forms*. Candidates include case-based/analogy-based estimation, neural nets, system dynamics, and use of new sizing, complexity, reuse, or volatility parameters.
- *Maintaining compatibility across multiple classes of models.* Including compatibility of inputs, outputs, and assumptions.
- *Total cost of ownership estimation*. In addition to software development, this can include estimation of costs of installation, training, services, equipment, COTS licenses, facilities, operations, maintenance, and disposal.
- *Benefits and return on investment estimation*. This can include valuation of products, services, and less-quantifiable returns such as customer satisfaction, controllability, and staff morale.
- *Accommodating future engineering trends*. These can include ultra large software-intensive systems, ultrahigh dependability, increasingly rapid change, massively distributed and concurrent development, and effects of computational plenty, autonomy, and biocomputing.

Given the dependence on the synergies between communities within the field of parametrics there are considerable sustainment challenges:

- *Sustained government leadership.* Through industry improvement incentives, contextualized data collection which evolves to accommodate needs. Sustained research sponsorship.
- *Sustained partnerships.* User groups such as IFPUG and cost model specific conferences must be maintained. Government/industry/academic partnerships such as NASA's Software Engineering Laboratory (SEL) in the U.S., National Information and Communications Technology Australia (NICTA), and Empirical Approach to Software Engineering (EASE) in Japan must continue to push the envelope for new research.
- *Professional society memberships.* Key to sustainment is the flow of new people into the field of parametrics. Similar challenges related to workforce aging, as observed in the U.S. aerospace and defense industries, exist in the field of parametrics.

These challenges contribute to the dynamic needs of the parametrics community, and provide a useful guide for the future needs.

Conclusions

Thomas Jefferson is credited with saying "History, by apprising [people] of the past, will enable them to judge the future." By exploring the history of parametrics, as told by the pioneers of the field, stakeholders can better determine how to advance the field to an unprecedented level of sophistication. It is evident from the experience of the pioneers that this is not done by a single stakeholder community. The symbiotic relationship between principal stakeholders is essential to the evolution and sustainment of the field. If the first 40 years are any indication, the field of parametrics will need to follow suit in order to make an impact.

The pioneers are optimistic about the future as the role of parametrics continues to be a critical aspect of government acquisition, increased collaboration takes place between corporations through professional societies, and research and education increases its profile in universities worldwide. Ultimately, these activities will keep the parametrics community in a highly stimulating and challenge-driven state.

Acknowledgements

This work is a reflection of the support and enthusiasm of the pioneers of parametrics. In particular, Barry Boehm has been my mentor throughout this process and the single largest influence for my continued involvement in the field of parametics. This work was funded by the National Science Foundation, the USC Center for Systems & Software Engineering, and the MIT Lean Aerospace Initiative.

References

Abdel-Hamid, T., Madnick, S., *Software Project Dynamics*, Prentice Hall, 1991.

Albrecht, A. J., & Gaffney, J., "Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation," *IEEE Transactions on Software Engineering*, Vol. SE-9, pp. 639-648, 1983.

Bailey, J. W. and Basili, V. R., "A Meta-Model for Software Development Resource Expenditures," *5th International Conference on Software Engineering*, San Diego, CA, pp. 107-116, 1981.

Boehm, B. W., *Software Engineering Economics*, Prentice Hall, 1981.

Doty Associates, Inc., *Software Cost Estimates Study*, Vol. 1, RADC TR 77-220, June 1977.

Gerlich, R., and Denskat, U., "A cost estimation model for maintenanace and high reuse," presented at *Proceedings, ESCOM*, Ivrea, Italy, 1994.

Hayes, W., and Zubrow, D., "Moving on Up: Data and experience doing CMM-based process improvement," CMU/SEI-95-TR-008 1995.

International Software Benchmark & Standards Group, "Data R8," 2005.

Jorgensen, M., and Shepperd, M., “A Systematic Review of Software Development Cost Estimation Studies,” *IEEE Trans. on SW Engr.*, Vol. 33, No. 1, pp. 33-53, 2007.

Madachy, R., "System Dynamics Modeling of an Inspection-Based Process," *20th International Conference on Software Engineering*, ACM/IEEE, 1996.

Madachy, R., *Software Process Dynamics*, IEEE-CS Press, 2007 (to appear).

Menzies, T., Chen, Z., Port, D., and Boehm, B., "Reduced-Parameter Estimation Models," presented at *Proceedings, PROMISE Workshop, ICSE*, 2005.

Mukhopadhyay, T., Vincinanza, S., Prietula, M., "Examining the Feasibility of a Case-Based Reasoning Model for Software Effort Estimation," *MIS Quarterly*, Vol. 16, pp. 155-171, 1992.

Parikh, G., Zvegintsov, N., "The World of Software Maintenance," in *Tutorial on Software Maintenance*, IEEE CS Press, pp. 1-3, 1993.

Putnam, L., "A General Empirical Solution to the Macro Software Sizing and Estimating Problem," *IEEE Trans. on SW Engr.*, pp. 345-361, 1978.

Ruhe, M., Jeffery, R, and Wieczorek, I., "Cost Estimation for Web Application," *27th International Conference on Software Engineering*, ACM/IEEE, 2003.

Royce, W., *Software Project Management*, Addison, Wesley, 1998.

Rubin, H. A., “Macroestimation of software development parameters: The ESTIMACS system,” *SOFTFAIR Conference on Software Development Tools, Techniques and Alternatives*, Arlington, VA, IEEE Press, pp. 109-118, 1983.

Selby, R., "Empirically analyzing software reuse in a production environment," in *Software Reuse: Emerging Technology*, W. Tracz (Ed.), IEEE-CS Press, pp. 176-189, 1988.

Shepperd, M. and Schofield, C., "Estimating Software Project Effort Using Analogies," *IEEE Trans. on SW Engr.*, Vol. 23, pp. 736-743, 1997.

Stutzke, R., *Estimating Software-Intensive Systems*, Addison Wesley 2005.

Walston, C. and Felix, C., "A Method of Programming Measurement and Estimation," *IBM Systems Journal*, Vol. 16, No. 1, 1977.

Wittig, G., "Estimating Software Development Effort with Connectionist Models," Monash University Paper 33/95, 1995.

Wolverton, R., "The Cost of Developing Large Scale Software," *IEEE Transactions on Computers,* Vol. 23, No. 6, 1974.

APPENDIX A. Interview Questions

Interview questions for Pioneers of Parametrics project

Project description: This study focuses on the impact of software engineering practices. We are particularly interested in exploring the genesis of the software resource estimation (SRE) field and its evolution between academia, government, and industry.

1. What got you interested in software resource estimation?

2. What kind of support did you have for developing your SRE capabilities? Did this constrain the scope of your SRE capabilities?

3. How had you previously been doing SRE?

4. What problems with you current approaches were you trying to address?

5. How did you determine the functional forms for your resource estimating relationships (RERs)?

6. To what extent did you draw on previous SRE models or studies of software cost and schedule drivers?

7. Which studies or models were most helpful?

8. What have been the most distinguishing features of the resource estimation relationships (RER) you developed?

9. Where did you get data to calibrate/validate the RERs?

10. Did the data and calibration cause any changes in your RERs?

11. Were there any environments that were difficult to model with your RERs?

12. What kinds of organizations initially and eventually were major users of your RERs?

13. What kinds of impacts did your RERs have on their practice?

14. How did your RERs evolve after their initial introduction?

15. What are the current and future challenges for the SRE field? Opportunities?

16. Knowing what you know now, would you approach SRE and RER development any differently?

17. What other individuals do you recommend we contact to interview on this topic?

Please draw a diagram showing what previous work influenced your model and what work your model has influenced.

APPENDIX B. Conferences, Journals, and Books Significant to Parametrics

Conferences

Association for Computing Machinery Special Interest Group on Metrics
Forum on COCOMO and Systems & Software Cost Modeling
Galorath International User Conference
IEEE International Symposium on Empirical Software Engineering and Measurement
International Function Point Users Group
International Society of Parametric Analysts
PRICE Systems International Symposium
Society of Cost Estimation and Analysis

Journals

Cost Engineering Journal
IEEE Transactions on Software Engineering[7]
Journal of Cost Analysis and Management
Journal of Cost Management
Journal of Empirical Software Engineering
Journal of Parametrics
Journal of Systems and Software

Books

Boehm, B., Software Engineering Economics, Prentice Hall, 1981.

Boehm, B. W., Abts, C., Brown, A. W., Chulani, S., Clark, B., Horowitz, E., Madachy, R., Reifer, D., & Steece, B., *Software Cost Estimation with COCOMO II*, Prentice Hall, 2000.

Conte, S. D., Dunsmore, H. E., & Shen, V.Y., *Software Engineering Metrics and Models*, Benjamin/Cummings Publishing Company, Menlo Park, 1986.

Galorath, D., and Evans, M., *Software Sizing, Estimation, and Risk Management*, Auerbach Publications, Boston, MA, 2006.

Gilb, T., *Software Metrics*, Winthrop Publishers, Cambridge, MA, 1977.

Jones, C., *Applied Software Measurement: Assuring Productivity and Quality*, McGraw Hill, New York, NY, 1991.

Jones, C., *Estimating Software Costs*, 2nd Ed., McGraw-Hill Osborne Media, 2007.

Kitchenham, B., & Littlewood, B., *Measurement for Software Control and Assurance*, Elsevier, 1989.

Kruchten, P., *The Rational Unified Process*, 2nd ed: Addison Wesley, 2001.

Nelson, E. A., *Management Handbook for the Estimation of Computer Programming Costs*, AD-A648750, Systems Development Corp., 1966.

Putnam, L. H., Myers, W., *Five Core Metrics: The Intelligence Behind Successful Software Management*, Dorset House Publishing Company, 2003.

Reifer, D., *Practical Software Reuse*, Wiley, 1997.

Royce, W., *Software Project Management*, Addison Wesley, 1998.

Stutzke, R., *Estimating Software-Intensive Systems*, Addison Wesley, 2005.

[7] listed as the most influential journal in software cost estimation by Jorgensen & Shepperd (2007)

APPENDIX C. Acronyms

COCOMO	Constructive Cost Model
COSMIC	Common Software Measurement International Consortium
COTS	Commercial off the shelf
DoD	Department of Defense
ESTIMACS	Estimator for Management and Computer Services
IEEE	Institute of Electrical & Electronics Engineers
IFPUG	International Function Point Users Group
MoD	Ministry of Defense (United Kingdom equivalent of DoD)
PRICE S	Parametric Review of Information for Costing and Evaluation Software
RUP	Rational Unified Process
SEER-SEM	System Evaluation and Estimation of Resources Software Engineering Model
SLIM	Software Life-cycle Model
SLOC	Source Lines of Code
SPQR	Software Productivity Quality and Reliability