

Technical Discussion Meeting
IEE Savoy Place, London
21 January 1999

M M Lehman

Dept. of Computing
Imperial College
London, SW7 2BZ
mml@doc.ic.ac.uk
<http://www-dse.doc.ic.ac.uk/~mml>

J F Ramil

Dept. of Computing
Imperial College
London SW7 2BZ
jcf1@doc.ic.ac.uk
<http://www-dse.doc.ic.ac.uk/~jcf1>



EPSRC funded project (Oct-96 to Sep-98) asking:

- Are there **similarities** in the long term **evolutionary behaviour** of different software systems?
- If yes,
 - what **common phenomena** do they reflect?
 - are there **practical implications** for
 - software evolution ?
 - its management and **control**?



Antecedents

- Pre 1994 studies limited to
 - OS/360 and other 70s studies (Lehman 69-84, Lehman & Belady 85)
 - Critical analysis of Lehman work (Lawrence, ICSE 6)
 - ICL VME evolution study (Kitchenham 92)
- **Multi-release** software evolutionary **behaviour** studies few and far between
 - Process highly dependent on **humans** thought unlikely to display disciplined behaviour over extended periods
 - **Context dependency** make common behavioural mechanisms over different applications and evolution environments unlikely
 - Potential of such studies not appreciated



The Laws (Pre 1994)

- Lehman studies led to the Laws of Software Evolution
- Encapsulation of observed behaviour

No.	Brief Name	Law
I	Continuing Change	<i>E</i> -type systems must be continually adapted else they become progressively less satisfactory.
II	Increasing Complexity	As an <i>E</i> -type system is evolved its complexity increases unless work is done to maintain or reduce it.
III	Self Regulation	Global <i>E</i> -type system evolution processes are self regulating.
IV	Conservation of Organisational Stability	The average effective global activity rate in an evolving <i>E</i> -type system tends to remain constant over the product lifetime.
V	Conservation of Familiarity	On average, incremental growth tends to remain constant or to decline.
VI	Continuing Growth	The functional content of <i>E</i> -type systems must be continually increased to maintain user satisfaction over their lifetime.
VII	Declining Quality	The quality of <i>E</i> -type systems will appear to be declining unless they are rigorously maintained and adapted to operational environment changes.



1994 - The FEAST hypothesis

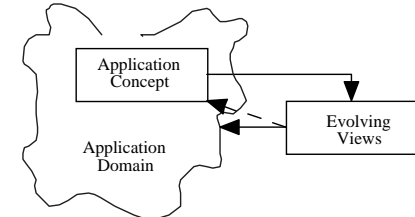
- **E-type global** software evolution processes are **multi agent, multi level, multi loop feedback** systems...
 - Observation dates back to OS/360 study and 1972 paper
 - **E-type** software:
 - **embedded** in and **implementing** an application in a **real world** domain
 - **user satisfaction** ultimate criterion of success
 - **Global** software process:
 - includes activities of all involved: developers, managers, executives, marketing, support personnel, users etc

Feedback loops...



Feedback Loops at the Step Level

- Initial, usually **implicit**, step 1
- E-type program new development or enhancement involves
 - **bounding** of **application concept** and **domain**
 - reflects and blends views of **many** stake holders

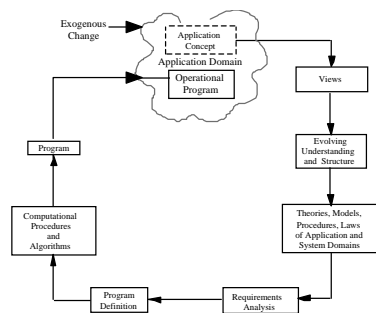


- An **iterative** process that changes **application concept** and **domain** bounds



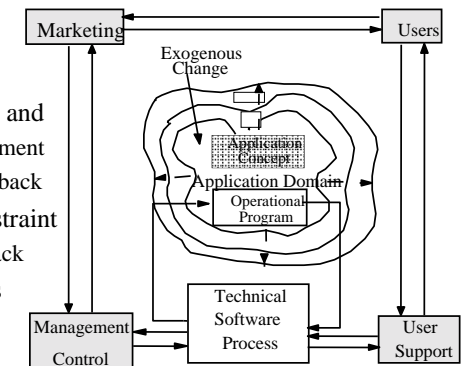
At the Release Level

- Technical-process steps lead to valid software **ready** for operation
- **Installation** and **operation** of completed system **changes** application domain
- Real world also undergoes changes
- **Rate** of change accelerated by installation and use of software
- Software contains **model** of its own operation
 - Software as a model **incomplete**
 - Gap implies embedded **assumptions**
- **unending software maintenance, that is, evolution**



Global Process - Outer Loops

- Technical process embedded in a wider process
- Many feedback loops
- Changing, expanding application and domain leads to continuing enhancement
 - positive, growth promoting, feedback
- **Growing complexity** major constraint
 - negative, growth limiting, feedback
- Management checks and balances



FEAST/1 Project (1996-98)

- Focused on **FEAST hypothesis**
- Evolution of several software systems studied:
 - British Aerospace Weapon System
 - ICL VME Operating System Kernel
 - Logica FW Financial Transaction System
 - Lucent Tech. Large real time system - 2 variants
- Compared to initial OS/360 study
- Data relates to evolution over periods of from 5 to 15 years
- Software technology of 60s to 90s
- **Different** processes and domains
- Approaches: - **black** and **white** box (system dynamics)



21-Jan-99

9

Some FEAST/1 results

- Strong **similarities** in growth trends of different systems
- **Black box** and **system dynamics** models of software evolution
- **Support** for laws of software evolution - minor refinements
- **Support** for hypothesis
 - observation
 - data interpretation
 - dynamic models
- **Seeds** for development of software evolution planning, management and **control** - tools
- Improved insight into **metrics of software evolution**
 - definition, collection, analysis



21-Jan-99

10

Current Practice - Metrics

- emphasis on measurement of *new* software
- collect rather than analysis
- metrics initiatives ‘from scratch’
 - tend to require substantial effort/commitment/cost from those involved in collection
- intense debate
 - what are the most *appropriate* metrics?
 - how to validate them?
- focus on local mechanisms
 - eg testing, inspections
 - local effectiveness



21-Jan-99

11

Evolutionary Behaviour

- Evolutionary behaviour relates to attributes of software **process**, its **product** and relevant **domains**
- Reflected in a set of **attributes**
 - Functionality
 - System Size
 - Effort applied
 - Quality
 - Reliability
 - Complexity
 - Changeability
- Described by **changes** in these attributes and **measures** of such changes



21-Jan-99

12

Metrics of Software Evolution

Some Desired Features...

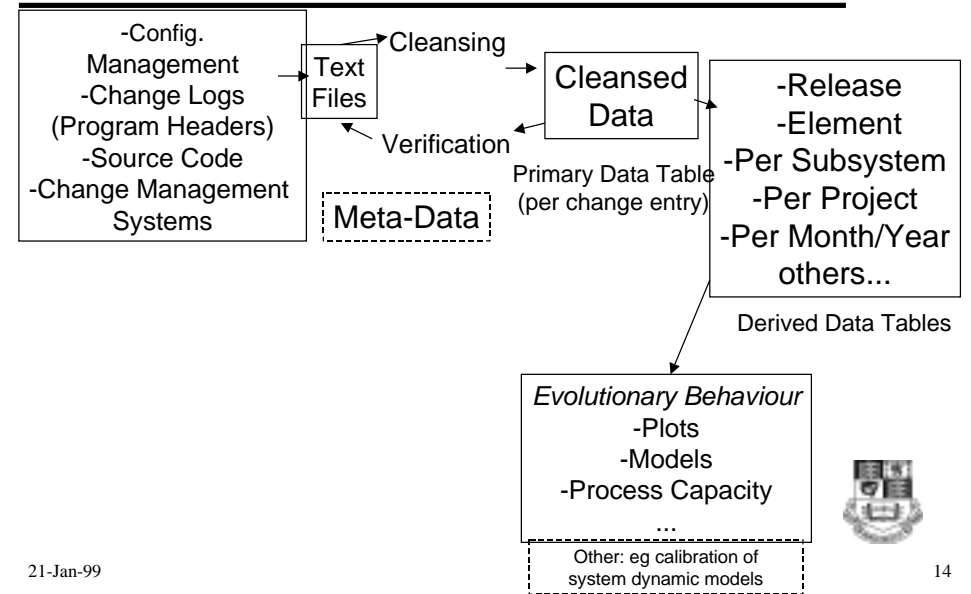
- Small set of metrics
- Representing long term evolutionary behaviour
- Readily or (almost readily) available from historical records
 - eg Configuration Management Databases
- Inexpensive to collect
 - minimising human intervention
- Focusing on *high level* process/product attributes
 - global process and product behaviour
- Enabling comparison across systems
- Refinement and more detail
 - as required by interpretation and analysis



21-Jan-99

13

VME Kernel HHR - The detailed view



21-Jan-99

14

Example

- Consider just two attributes
 - **size** - product attribute
 - number of elements (modules, files, etc)
 - **handled** - process attribute
 - number of **added** elements
 plus
 - number of different elements **modified** within a given period of time
 or within one release interval (from release n to release n+1)

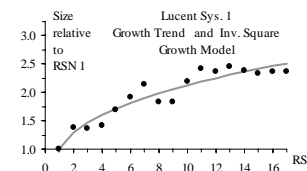


21-Jan-99

15

Product Size - Growth Trends

- Inverse Square Growth - an indicator of increasing complexity
- Good prediction power.
- Only a few points needed to estimate the model.



Inverse Square Growth Model (ISM)

$$S_i = S_{i-1} + E / (S_{i-1})^2$$

$$i = 2, \dots, n$$

S_i is predicted size of system at rsn i
 E , for example, as average E_i
 $E_i = (y_i - y_1)_{k=1} \sum_{k=1}^{i-1} (1/y_k^2) \quad i = 2, \dots, n$



21-Jan-99

16

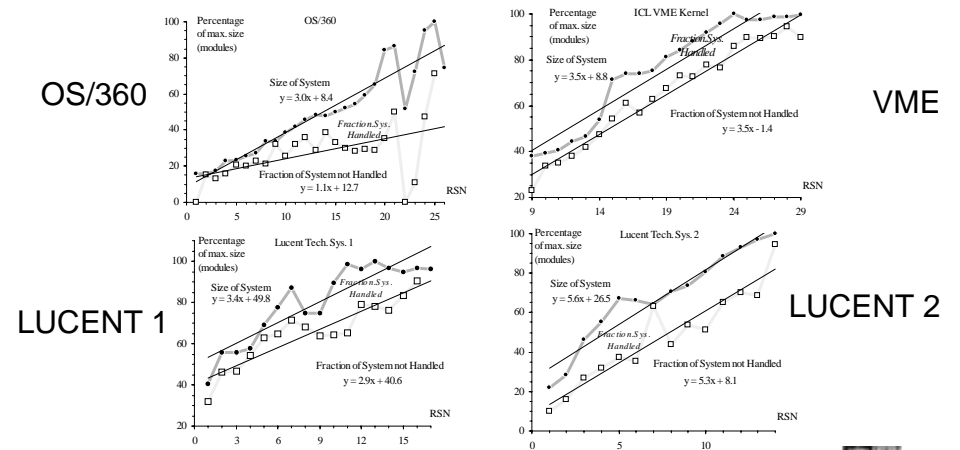
Prediction Power of the Inverse Square Model Several Systems

System	Model	%MAE	St. Dev. of % Residuals
OS/360-370-rsn 1-19	Linear.	4.7	5.9
OS/360-370-rsn 1-26	Linear.	9.6	14.0
OS/360-730-rsn 1-19	Inv. Sq.	20.1	25.0
OS/360-370-rsn 1-26	Inv. Sq.	30.9	36.7
VME Kernel - rsn 1-13	Inv. Sq.	8.0	11.4
VME Kernel - rsn 14-29	Inv. Sq.	3.7	5.1
FW	Inv. Sq.	3.6	4.8
Lucent Sys 1	Inv. Sq.	6.0	7.2

•% MAE - Mean Absolute Error in Percent



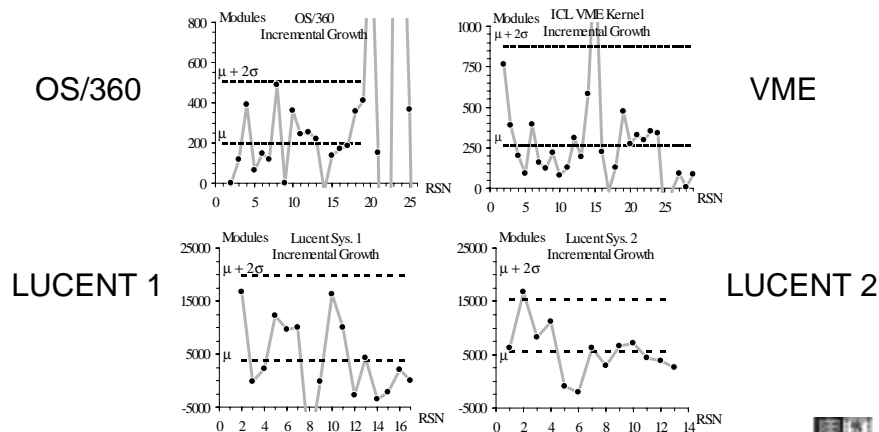
Fraction of System Handled and Stable Growth



•Fraction of system handled constrained to some maximum for stable growth



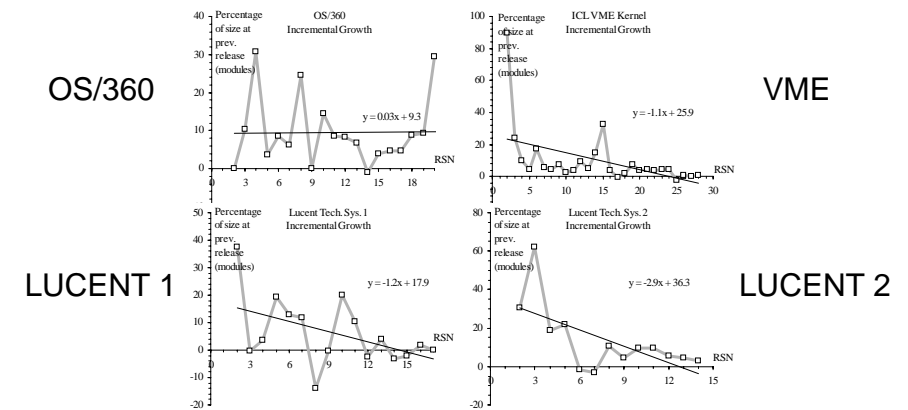
Limit Infringement in Incremental Growth



•Incremental growth greater than some threshold - followed by decline in subsequent releases



Declining Incremental Growth



•Non due to declining need or demand
•Non declining incremental growth in OS/360 - precursor of instability



The Laws (1998)

No.	Brief Name	Law
I	Continuing Change	<i>E</i> -type systems must be continually adapted else they become progressively less satisfactory.
II	Increasing Complexity	As an <i>E</i> -type system is evolved its complexity increases unless work is done to maintain or reduce it.
III	Self Regulation	Global <i>E</i> -type system evolution processes are self regulating.
IV	Conservation of Organisational Stability	The average effective global activity rate in an evolving <i>E</i> -type system tends to remain constant over the product lifetime.
V	Conservation of Familiarity	On average, incremental growth tends to remain constant or to decline.
VI	Continuing Growth	The functional content of <i>E</i> -type systems must be continually increased to maintain user satisfaction over their lifetime.
VII	Declining Quality	The quality of <i>E</i> -type systems will appear to be declining unless they are rigorously maintained and adapted to operational environment changes.
VIII	Feedback System	<i>E</i> -type evolution processes constitute multi-level, multi-loop, multi-agent feedback systems and must, in general, be treated as such to achieve significant process improvement for other than the most primitive processes.



21-Jan-99

21

Implications for the Control of Software Evolution

- An explanation for
 - **unending** software maintenance
 - difficulty in achieving **global** process improvement
 - disciplined evolutionary behaviour
- Limited global effect of local improvement
 - constraining effects of feedback mechanisms
 - they must be considered when seeking global improvement
 - **forward** paths changes demand changes to **feedback** mechanisms
 - improvement has to be measured from outside
- Feedback must be **managed**

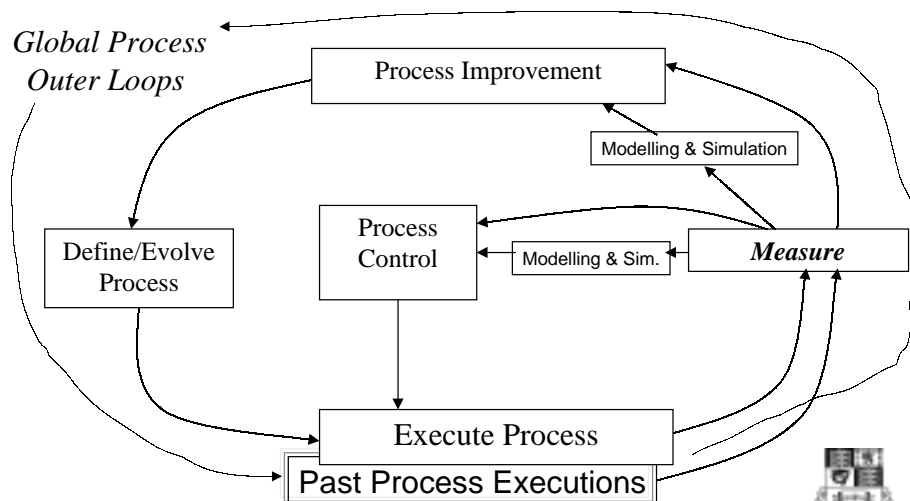


21-Jan-99

22

Metrics in Software Evolution Management

from (Christie, 98), adapted



21-Jan-99

•Process Control under the effect of Global Process Loops

23

Further Work

- Metrics-based framework for the control of software evolution
 - ‘primitive’ metrics & ‘data rich’ processes
 - tooling and automation for data extraction
- Estimation and planning models for evolving systems
 - fuzzy dynamics, multi-stage fuzzy control (Kacprzyk, 97)
 - ‘maintenance projects’ and ‘sequential experimentation’
- Validation - Study of additional systems - further aspects
 - Identify, explain commonalities of behaviour
 - ‘Inherited’ dynamics, role of application domain, short-term behaviour
- FEAST/1 project just a first attempt. There is a need for
 - multidisciplinary approaches
 - more widespread investigation



21-Jan-99

24

Some Conclusions

- Similarities over 30 year period indicate **underlying phenomena** beyond software technology
- Project demonstrates **presence** and **impact** of feedback on global process
- Further investigations required - FEAST/2 starting soon
- Contribution towards **theory** and **practice** of software evolution



Some references

- [Chr98] Christie A., *Simulation in Support of Process Improvement*, paper and charts, ProSim'98, Int. Workshop on Software Process Simulation and Modelling, June 22-24, 1998., Silver Falls, Oregon. (Selected papers will appear in a special issue of Journal of Sys. & Softw. 1999)
- [Kac97] Kacprzyk J, *Multistage Fuzzy Control*, John Wiley & Sons, Chichester, 1997
- [Kem98] Kemmerer C. and Slaughter S., *A Longitudinal Empirical Analysis of Software Evolution*, Proc. IWPE'98 Int. Workshop on Principles of Softw.Evolution (ICSE'98), April 20-21, Kyoto, pp. 21-28
- [Kit82] Kitchenham B., *System Evolution Dynamics of VME/B*, ICL Tech. J., May 1982, pp. 42-57
- [CDP98] *Code Decay Project* (<http://www.bell-labs.com/org/11359/projects/decay/biblio.htm>)
- [Leh85] Lehman MM and Belady LA, *Program Evolution - Processes of Software Change*, Ac.Pr.,1985
- [Leh97]] Lehman MM, Perry DE and Ramil JF, TurSKI WM, and Wermick P, *Metrics and Laws of Software Evolution - The Nineties View*, Proc. Metrics 97, Albuquerque, NM, 5-7 Nov 97, pp 20-32
- [Leh98a] Lehman MM, *FEAST/2 Project: Case for Support*, Dept. of Comp., Imp. Col., July 1998
- [Leh98b] Lehman MM, Perry DE and Ramil JF, *Implications of Evolution Metrics on Software Maintenance*, Proc. Int. Conf. on Soft. Maint. (ICSM'98), Washington DC, Nov.16-24, 1998
- [Leh98c] *id.*, *On Evidence Supporting the FEAST Hypothesis and the Laws of Software Evolution*, to appear, Proc. Metrics'98, Bethesda, MD, Nov. 20-21,1998
- [Leh98d] Lehman MM and Ramil JF, *Implications of Laws of Software Evolution on Continuing Successful Use of COTS Software*, Dept. of Comp., Imp. Col., Res. Report 98/8, submitted to publication, 1998

FEAST/1 references available via:

<http://www-dse.doc.ic.ac.uk/~mml/feast>

