

ACCESS

Autonomic Computing - Creating self-Evolving Software Systems

Tim Millea
Applied Software Engineering Research Group
University of Reading
tim@millea.com

ACCESS Tim Millea,

ACCESS

- 1.introduction
- 2.perspective - what am I talking about?
- 3.evolutionary systems
- 4.evolution
- 5.autonomic computing
6. ACCESS

ACCESS Tim Millea,

background

- Durham BSc Computer Science (1992-96)
 - software engineering / AI
- Oxford University MSc in Computation (1996-7)
 - project: intelligent agents for the web
 - parallel close-coupled evolutionary algorithm
- Loughborough PhD 'Immortal Software' (1997-2001)
 - Evolutionary Financial IT project with Schroders Bank
- Reading University
 - Research Fellow on DESEL (2001- 02)
 - Lecturer in Computer Science (2002 - 04)
 - First Grant Proposal - ACCESS
- Next - Antalya, Turkey
 - Property developer, property management, sales

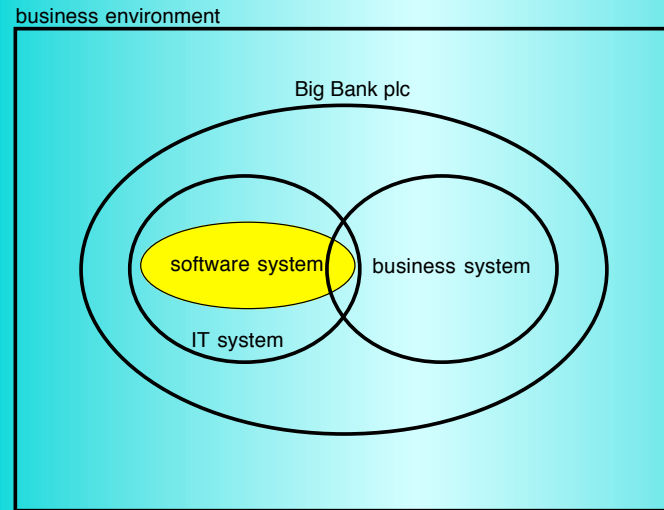
ACCESS Tim Millea,

ACCESS - facts

- Autonomic Computing - Creating self-Evolving Software Systems
- £113 000 grant (TL 300 billion)
 - £4k equipment, £9k travel, £100k for 2 studentsh
- First started October 2003, ends April 2007
- Project personnel:
 - Principal Investigator: Mike Evans
 - Co-Investigator: Rachel Harrison
 - Specification, DSLs: Diana Griffiths
 - Software markets: Tim Putnam

ACCESS Tim Millea,

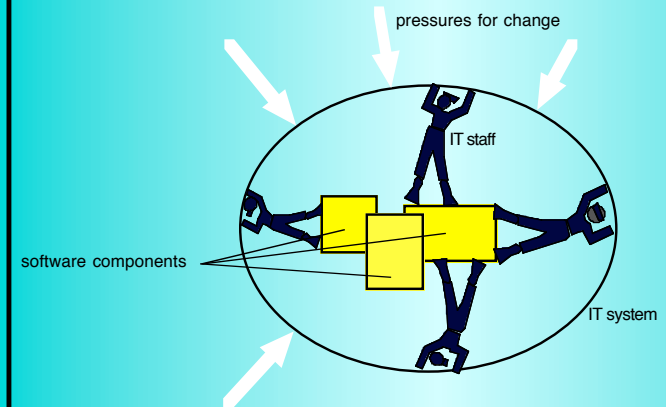
perspective



ACCESS Tim Millea,

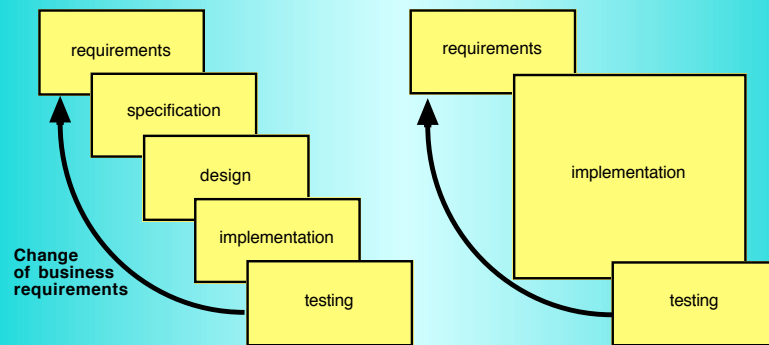
IT System = software + people

IT staff as shock absorbers in an IT system - between the pressures for change and inflexible software



ACCESS Tim Millea,

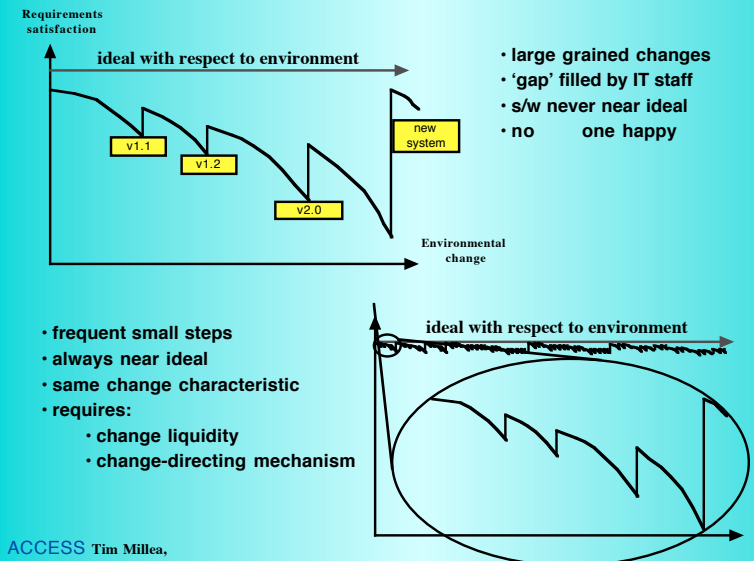
poor feedback



- poor customer-developer feedback loop
- customers and developers live in different worlds

ACCESS Tim Millea,

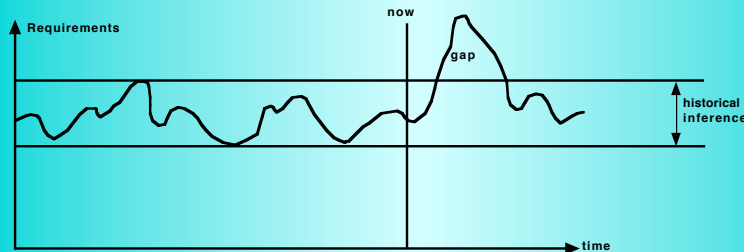
the requirements gap



ACCESS Tim Millea,

building-in flexibility / predicting the future

- historical inference
- use domain knowledge
- build upon static foundations



- structure vs flexibility
- align the structure with the dynamics of the domain

ACCESS Tim Millea,

aligning the dynamics

- hypothesis: every domain has its own set of atomic constants
- e.g. in the financial domain: transaction, debt, equity, sale etc.
- by identifying atomic constants, we infer that everything else in the model may change
- customer constructs system not from programming primitives, but from domain primitives
- graduation from atomic constants to dynamic applications, aligns software & application domain dynamics

ACCESS Tim Millea,

shortening the feedback loop

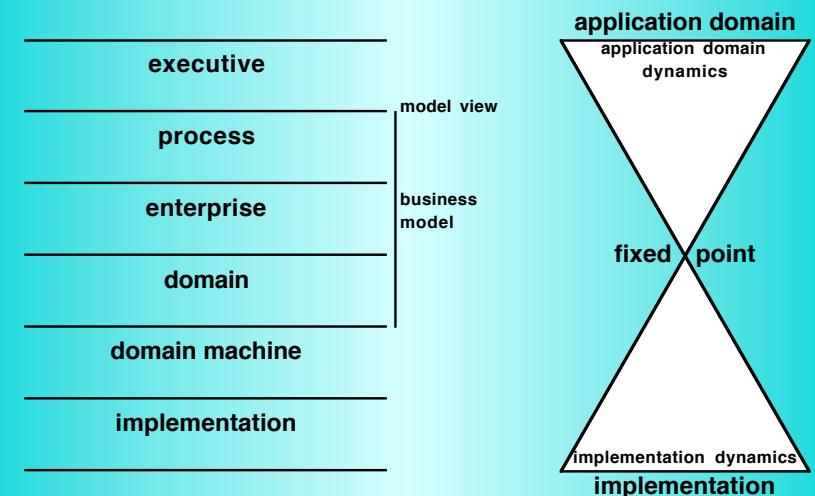
- raise level of system construction towards customer by automating development stages

Life cycle stage	Paradigm	Example
requirements	evolutionary	evolutionary algorithm
specification	semantic	executable model
design	component	component library
implementation	program	C, Pascal

- end user/customer becomes agent of change
- longer term ideal would be automatic evolution from requirements

ACCESS Tim Millea,

layered model of evolutionary systems



ACCESS Tim Millea,

alternatively - software market failures

- software evolution problems due to inefficient allocation of resource
- lack of regulation has led to:
 - 'software pushers' getting users hooked
 - forced upgrades
 - tied products
- regulation should provide
 - open standards
 - compatible OS and application frameworks
 - better competition
 - licenses meaningful to the user (eg. service standard & duration)
 - protection against upgrade coercion

ACCESS Tim Millea,

evolvability - the 3 Fs

Flexibility

- ability to change
- dimensions of change
- 'solution space'

Feedback

- directing change
- environmental interaction

Formality

- the 'rule book'
- maintaining correctness
- meta-structure
- limiting scope

ACCESS Tim Millea,

defining evolution

defn. evolution :

a process of feedback-driven (beneficial) change

$$E(s) \Leftrightarrow s_t = f(s_{t-1}, e_{t-1})$$

ACCESS Tim Millea,

why self-Evolving Software Systems?

- no. of computers growing approx 40% p.a
- average complexity increasing
- managing it is currently a human activity
- labour demand outstrips supply
- forecast 200M skilled personnel required within 10 years

UK e-Science autonomic computing research agenda

- self-organising adaptive systems
- managed through specified policies and agreements between system stakeholders
- interoperability across domains

ACCESS Tim Millea,

Autonomic Computing

autonomic - self-regulating functioning of central nervous system, e.g. heart beat rate, pupil dilation, perspiration, adjusted according to changing conditions and without conscious thought. By analogy,

autonomic computing systems:

- self-regulate
- self-maintain
- respond to changing requirements
- provide optimal service
- without end-user intervention.

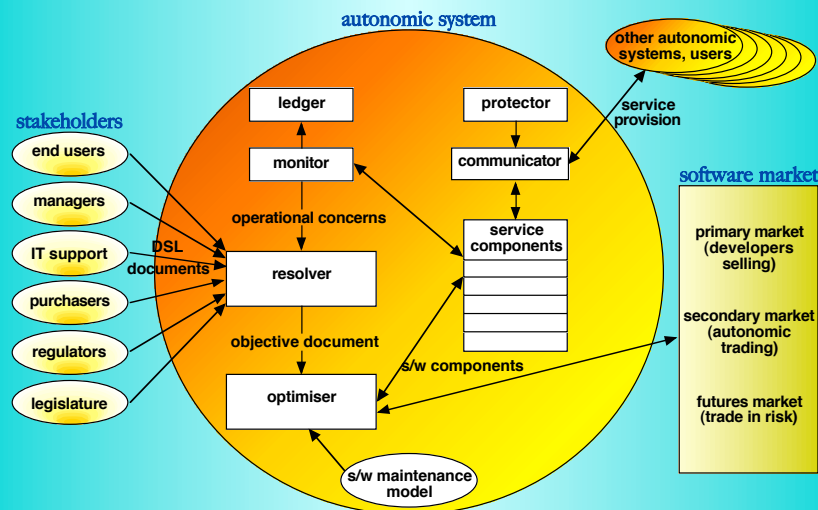
ACCESS Tim Millea,

IBM's 8 elements of autonomic computing

1. 'self-aware'
2. self (re)configuring
3. self-optimising
4. self-repairing
5. self-protecting
6. self-adapting
7. operate in heterogeneous world
8. hide internal complexity

ACCESS Tim Millea,

Autonomic Computing - Creating self-Evolving Software Systems



ACCESS Tim Millea,

deliverables

- Family of DSLs
- Domain calculus
- Tool to support the rapid generation of DSLs from domain calculus expressions.
- Theory of resolution and its application to deal with inconsistencies, uncertainties and compromise in expressed concerns.
- A model and implementation of a resource-efficient software market extending the service-oriented approach to machine trading.
- Evolutionary algorithm to optimise with respect to a set of resolved system concerns and the software market.
- Published output proving that we have created self-evolving software

ACCESS Tim Millea,

The Babel Fish

"The oddest thing in the Universe"

- Feeds on brainwave energy
- Excretes a telepathic matrix
- Combines thought & speech nerve signals
- Practically:
 - Stick one in your ear to understand any form of language
- "Nothing so mindbogglingly useful could have evolved purely by chance"



[From Hitch-Hikers Guide to the Galaxy", Douglas Adams, 1979]

ACCESS Tim Millea,

summary

- **background**
 - Durham to Reading, evolutionary algorithms to autonomic computing
- **perspective**
 - software, people, feedback, inflexibility, requirements gap
- **evolutionary systems**
 - layered & 'egg timer' models
- **evolution**
 - the '3 Fs', definition and equation - $\mathbf{E}(s) \leftrightarrow s_t = f(s_{t-1}, e_{t-1})$
- **autonomic computing**
 - meaning, research agenda
- **ACCESS**
 - TL 300 billion 3 year project started Oct 2003
 - funding for two PhD students and blue-sky conferences

ACCESS Tim Millea,

