

The future of computing and ITSM

Neil McBride
Centre for IT Service Management
Research
De Montfort University

Arguments

The computational model of computing is reaching its limits.

A lot of what we do in IT service Management is primitive.

New approaches are needed to create computer systems which are adaptive, self-maintaining and self-repairing.

Paradigms for such approaches need to come from other fields.

Biological Frameworks may provide useful metaphors.

New forms of computing will shift the function of IT service managers.

The focus will be on service, not IT.

A constantly changing world

Dynamic business environments.

Changing organisations.

Unpredictable economic conditions.

Unexpected changes.

Temporary stable patterns of behaviour, positive feedback, unpredictable chaotic shifts in behaviour.

IT requires stability and predictability.

The world is not stable and predictable.

IT breaks and fails

Creating adaptability in a changing world

Limits of Computing

Computer systems are brittle,

Difficult to update software,

No awareness of context,

No understanding of boundaries,

Cannot deal with uncertainty (e.g. fuzzy logic),

No self-awareness,

Lousy user interfaces,

Very difficult to do general pattern recognition,

Doesn't exhibit homeostasis.

Generally de-humanising and inflexible.

Elements of IT Service

Problem management,
Availability management,
Asset Management,
Release and configuration management,
Security management,
Repairs and replacements,
Network management.

What's the common denominator?

What are we doing?

Managing the inadequacies of the technology;

Fighting a losing battle to limit complexity;

Serving the technology and ignoring the people;

Forcing the business to serve the IT rather than IT serving the business;

Doing the plumbing jobs and maintenance jobs that the computers should be intelligent enough to do themselves;

Focusing on the IT rather than the service.

We should be:

Designing and tailoring information services to support the business's services;

Letting the technology worry about its own repairs.

Computing systems need to be:

Self maintaining,

Automatically reconfigurable,

Self adapting,

Evolvable,

Self-developing.

But they're not!

The computational model is inadequate.

We need new models, new metaphors, new ways of building computers and running them.

Future Directions

The Feyerabend Project and Biological Framings for Problems in Computing;

Autonomic Computing;

Service-oriented Computing.

Why the Feyerabend Project?

Computing theory was first developed as a branch of computability theory based on a particularly inexpressive set of mathematical constructs.

Early computing practices evolved under the assumption that the only uses for computers were in military, scientific or engineering computation.

The architecture of almost every computer today is designed to optimise the performance of Fortran programs and its operating-system level sister, C.

Programming languages have hardly shown one scintilla of difference from the designs made in the first few years of computing.

Software development methodologies evolved under the mythical belief in master planning.

Every single programming language is predicated on the physicist's model of figure it out, code it up, compile it, run it , throw it away.

“Fifty years into the First Computing Era some of us in the computing arena have come to realize we’ve made a false start that can’t be fixed, and for us to finally be able to produce lasting, correct, beautiful, usable, scalable, enjoyable software that stands the tests of time and moral human endeavor, we need to start over. Perhaps we’ll be able to salvage some of what we’ve learned from the First Era, but I expect almost everything except the most mathematical fundamentals to be brushed aside.” (Dick Gabriel)

Paul Feyerabend *Against Method*

Science as an anarchic enterprise.

Anything goes.

Thinkers decide not to be bound by methodological rule or unwittingly break them.

'Given any rule, however 'fundamental' or 'necessary' for science, there are always circumstances when it is advisable not only to ignore that rule, but to adopt its opposite.'

Advancing science may involve proceeding counter-intuitively.

Demanding consistency locks science into well-confirmed theories.

Proliferation of theories is beneficial to science.

No idea is incapable of improving our knowledge.

Aims of the Feyerabend Project

Understand the limitation of our current computing paradigm.

Understand the limits of current development methodologies.

Bringing people into the design process.

Make programming easier by making computers to more of the work.

Use deconstruction to uncover marginalised issues and concepts.

Looking to other metaphors.

Biological Framing of Problems in Computing

Workshop at Santa Fe Institute <http://www.santafe.edu> :

Examining the limits of current computing paradigms,

Identifying Hilbert Problems in computer science,

Exploring the value of biological metaphors.

Biological Models

Genes, gene expression and gene control.

Application domains.

Homeotic genes.

Receptors and feedback.

Evolving generations.

Control networks: Expression control, signalling networks, developmental control.

Emergent properties.

Autonomic Computing

‘It’s time to design and build computing system capable of running themselves, adjusting to varying circumstances and preparing their resources to handle most efficiently the workloads we put upon them. These autonomic systems must anticipate needs and allow users to concentrate on what they want to accomplish rather than figuring out how to rig the computing systems to get them there.’ IBM

Autonomic Computing

The system will have self-awareness.

The system will be able to configure and reconfigure itself.

The system will be self-monitoring and will be constantly seeking self-optimisation.

The system will be self-healing able to recover from malfunctions.

The system will protect itself, identifying threats and reacting to them, developing protection against new viruses using an artificial immune system.

It will be sensitive to its environment, identifying neighbouring systems and working out how to interact with them.

The autonomic system will manage itself in an open environment, where global computing is the norm and connections between internal organisational systems and global networks are natural and expected. It will operate in a non-proprietary environment, connecting to other computing systems and resources as needed.

The autonomic system will anticipate IT resource requirements; drawing in extra resources or redistributing existing resources in response to anticipated business change.

Autonomic Computing

Difficult to build!

1. Making individual system components autonomic.
2. Achieving autonomic behaviour at the level of global enterprise IT systems.

The ideas are good, but the implementation may be years away.

How we do it may depend on drawing further inspiration from biology.

IBM eLiza Project

Machines protect themselves from hackers, repair their own problems as they happen and manage their own operations.

Based on software algorithms.

Enterprise Workload Manager.

‘First generation of eWLM won’t include automated control features such as the ability to change sizes of logical partitions when applications need more disk space... still probably 18 months away’

Tivoli

‘taking a first step towards monitoring tools that can fix a flaw once they find one.’

Service-Oriented Software

New thinking in the UK computer science community,
Delivering software as a service,
Focus on flexibility and agility.

Software:

- Meets necessary and sufficient requirements,
- Is personalised,
- Is self-adapting,
- Is structured in simple units,
- Is geographically and platform transparent.

Service-oriented Software

Software component assembly driven by service definition and demands.

Software is configured on the fly to meet a service need.

Software components aggregate in response to a service requirement and then disaggregate when that service is not required.

Software building is then dynamic and service-driven.

The service is the master of the software, not the software the master of the service.

Task becomes one of defining and controlling the service level agreement

Adaptable computers and the consequence for IT Service Management

From managing the technology to managing its use,

From IT services to business services,

From routine trouble-shooting to exceptional trouble-shooting,

From service operation to service strategy.

IT Service Role

Strategy

Definition

Design

Customisation

Adaptation

What service will suit this customer?

What will it look like?

What will go in the SLA?

IT services: Mediators between the business community and the adaptive system.