Presentation to BCS Advanced Programming SG

Brownfield Development - 8th December 2011

Richard Hopkins
IBM Distinguished Engineer FIET

Kevin Jenkins
IBM EXEC ITA FBCS FIET

Grady Booch
IBM Fellow

Chris Winter
IBM Fellow Emeritis FBCS FIET

The Brownfield Problem

The majority of large organisations, in both the private and public sectors, many of whom are IBM customers, have invested heavily in IT solutions, in some cases for over forty years.

This investment has left them with a very large, complex and difficult to change IT landscape.

In many ways these landscapes resemble Brownfield construction sites where the process of building or augmenting a property is a risky and expensive business.

Changes, especially major changes, have to be designed to accommodate what already exists often without documentation of, or available expertise for the existing elements.

After 40 years of investment, complex IT runs the world, but our customers need an immediate reduction of duplication and complexity

500 million years of programming investment

Brownfield Re-engineering

Brownfield Re-engineering will manage the development and deployment of new and/or modified capabilities within the complexity of today's existing IT systems. This will be achieved by augmenting today's Greenfield approaches by:

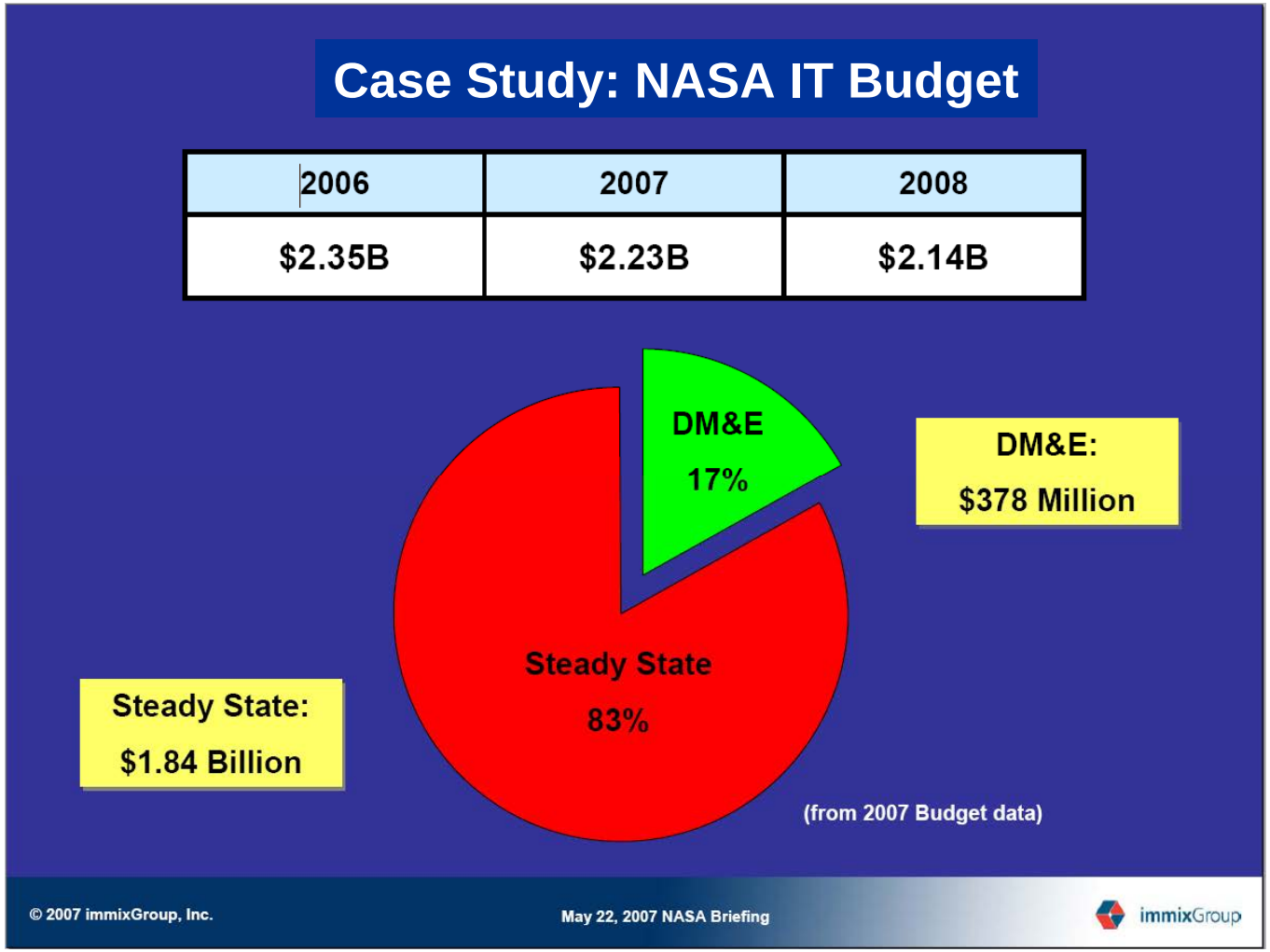
Conducting of Site Surveys that will harvest metadata. Storing this metadata in Inventories that provide a single source of the truth.

Abstractions of this metadata will provide higher level representations to assist with understanding the complexity.

Reliable forward engineering can then be performed from this solid engineering base.

Brownfield Engineering will improve the understanding & management of increasingly complex IT environments.

The Study has found that in the 1970's over 70% of the IT budget was spent on innovation. Over the last forty years that position has now been more than reversed. Major organisations, including IBM, spend more than 80% of their IT budget on the steady state.



The Brownfield IT site, therefore, is the reality of the vast majority of today's and tomorrow's IT landscapes. The evidence clearly shows that our current (and historic) approaches to contain IT complexity have been ineffective. To design, develop and re-factor an even more complex Smart Planet, we will need better techniques…

"A simple back of the envelope calculation suggests that, worldwide, we produce about 33 billion lines of new or modified code every year. Cumulatively, this means that since the 1940's and 50's we've produced somewhere around one trillion source lines of code… it's a humbling thought, for through those trillion lines of code… we've changed the world"

Grady Booch, Preface to *Eating the IT Elephant*, May 2008

"The moment you write a line of code, it becomes legacy. If the cumulative mass of that legacy is small, then there is correspondingly little inertia; if the cumulative mass is large, then there is considerable resistance to change. Refactoring becomes more and more critical as mass increases, because it drives a software-intensive system to intentional simplicity, thus creating a more frictionless surface."

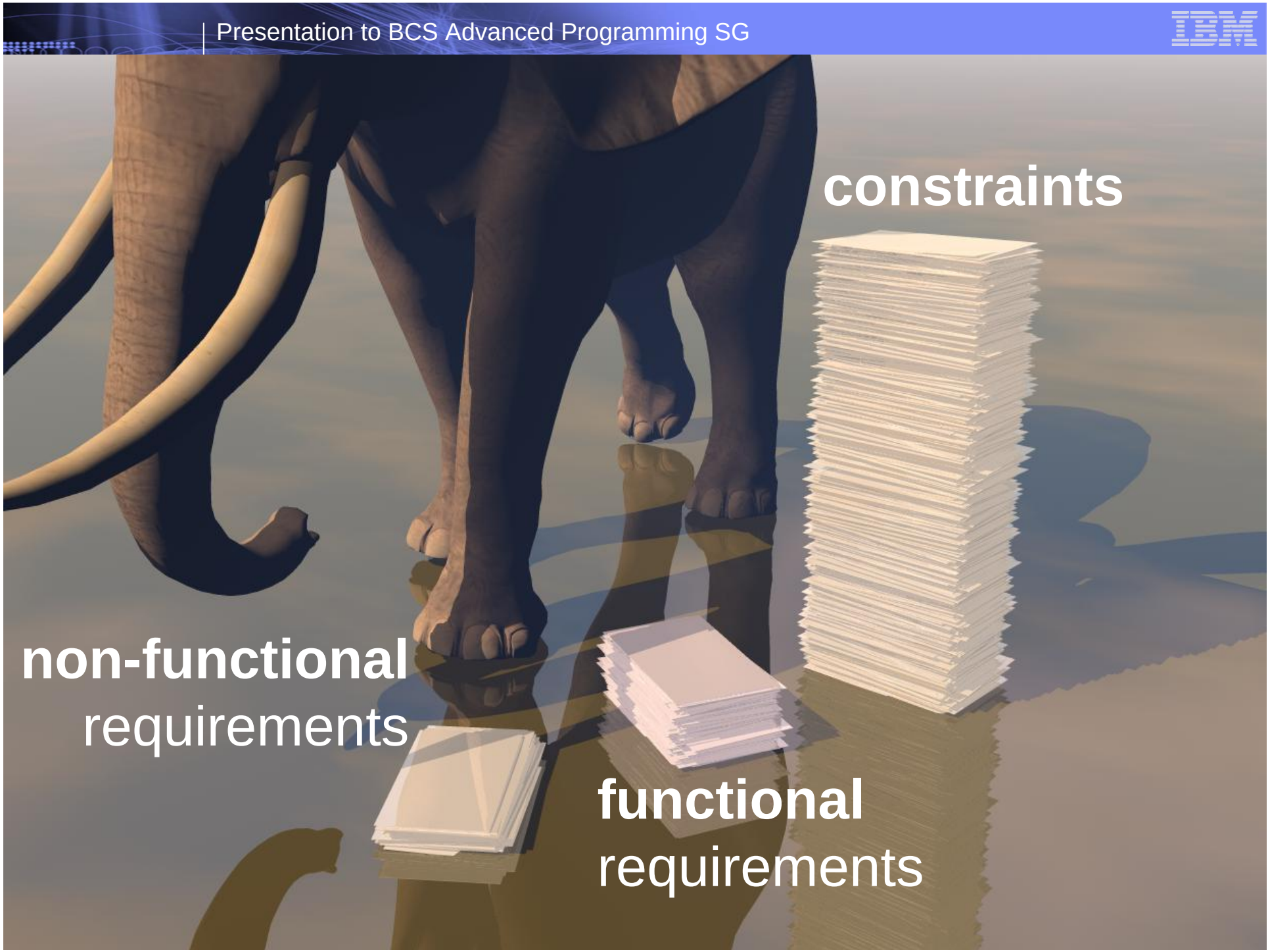
Grady Booch, Software Archaeology, *handbook of software architecture blog*, May 2008

"any system that has evolved over the years… will naturally have lots of inefficiencies. If we can identify those inefficiencies, through extensive information gathering and analysis, as well as attention to the workings of individual processes, we can then begin to reduce those inefficiencies. Over time we will have significantly optimized the functioning of the total system... How can we make better decisions given the increasingly complex problems all around us? Often, the reason these decisions are so difficult to make is because we just do not know what is going on. We may have a hunch, but hunches are not enough when treating a very sick patient or managing a financial system seemingly in free fall."

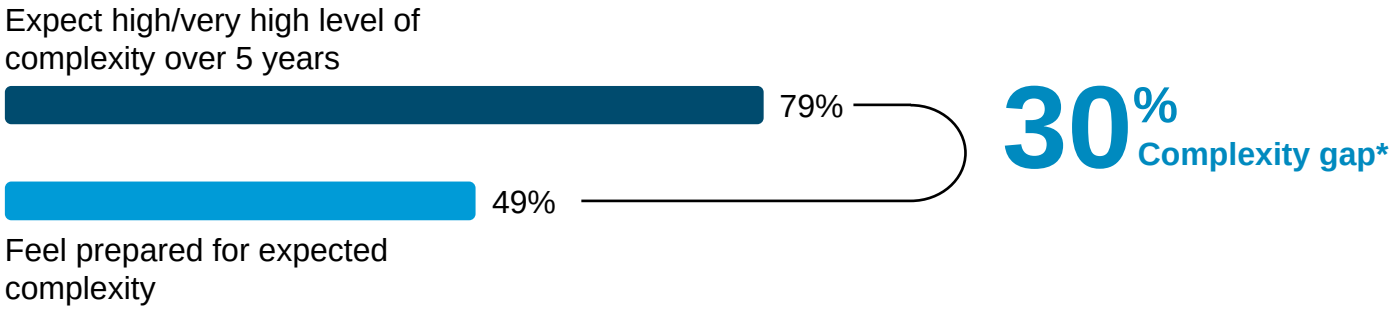
Irving Wladawsky-Berger, Living in a Smarter Planet, November 2008

"According to The Office of Management and Budget (OMB), the number of federal data centers grew from 432 in 1998 to more than 2,000 in 2010."

Testimony Before the Subcommittee on Financial Services and General Government, Committee on Appropriations, U.S. House of Representatives
March 2011


constraints
non-functional
requirements
functional
requirements

CEO's do not feel they are fully prepared to meet the complexity they will experience over the next five years…



- The IBM 2010 CEO Study provides the following advice to CEOs:
  - Simplify whenever possible
  - Actively manage systemic complexity
  - Promote a mindset of being fast and flexible
- CIOs and CTOs therefore need to identify new IT delivery approaches that can absorb increasing levels of systemic complexity and incrementally simplify it in a way that is fundamentally efficient and agile.
- Their System of systems are, almost without exception, evolved and integrated complex brownfield IT landscapes. To efficiently understand and engineer we need to expand our current thinking.

*Source: IBM CEO Study Report 2010

Problem Statement

- Understanding the **complexity** of **current** Business and IT Systems, consumes up to 68% of the effort of developing new business capabilities.
  - Current “Greenfield” tooling and methods are not sufficient to help free part of the 66% for new and innovative functionality.
  - Insufficient tooling for discovery and abstraction of existing system
  - Inconsistent understanding of complexity attributes
- The CHAOS survey from the Standish Group has tracked an overall improvement in IT project delivery success over the last twenty years, but even in 2006, large IT projects still failed more often than succeeded.

Case Study: Effort Breakdown by Major Category

Package Configuration and Customisation

The *Framework* comprises of functionality to enable two COTS packages to run in the client’s estate

600 *Notifications* in dual language (English and Welsh)

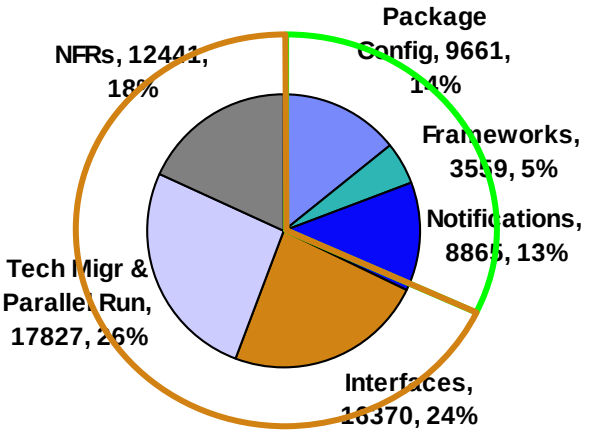
155 *Interfaces* to be developed

Technical Migration and Parallel Running is required from install to legacy decommissioning

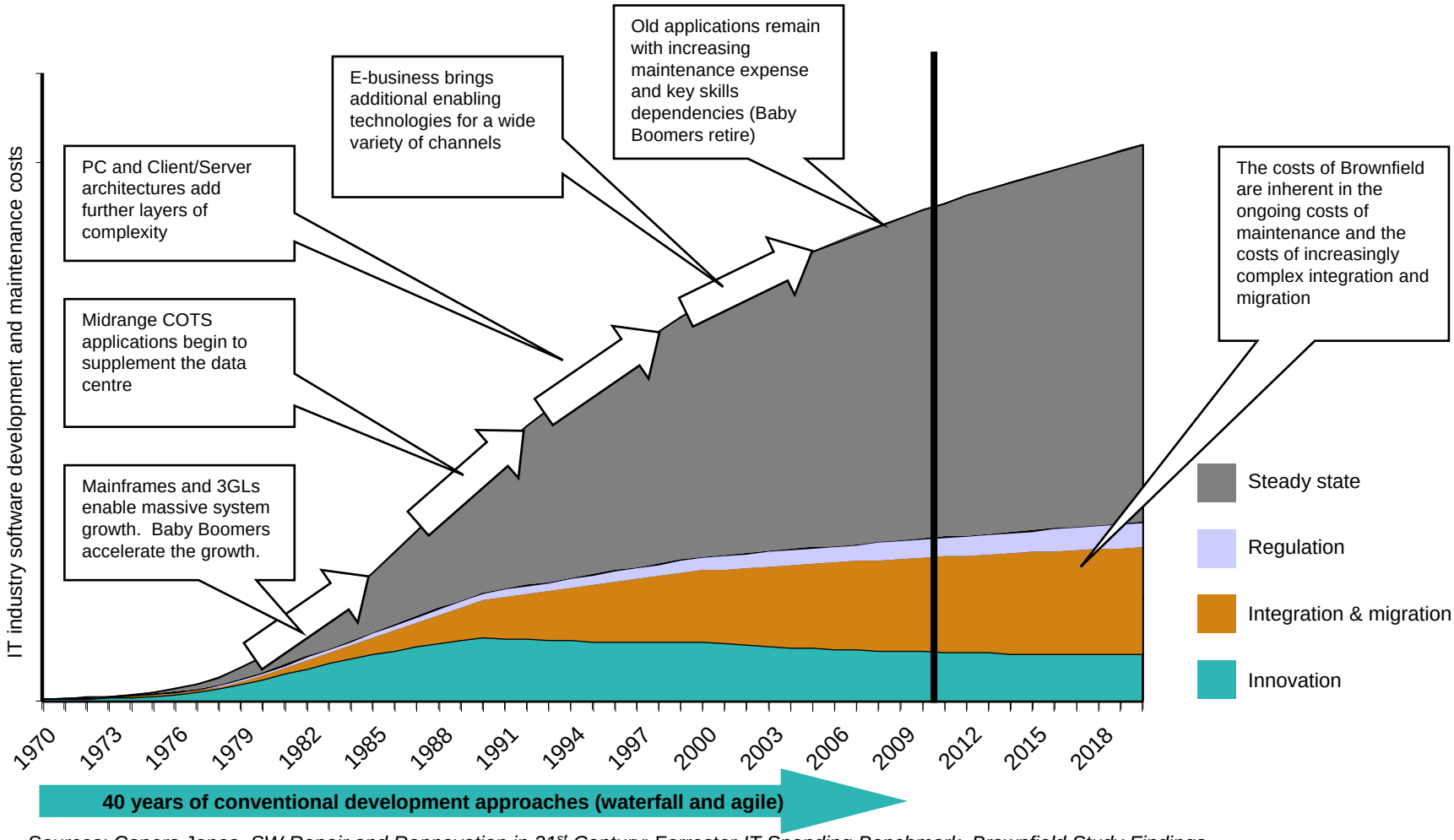
130 *Non Functional Requirements (NFRs)*, the Top 10 NFRs are 50% of build effort:
Performance, Availability, Auditability, Scalability, Security & Welsh Language

32% of the programme effort is configuration and customisation

68% of the programme effort is required to *Integrate the packages into the clients estate*

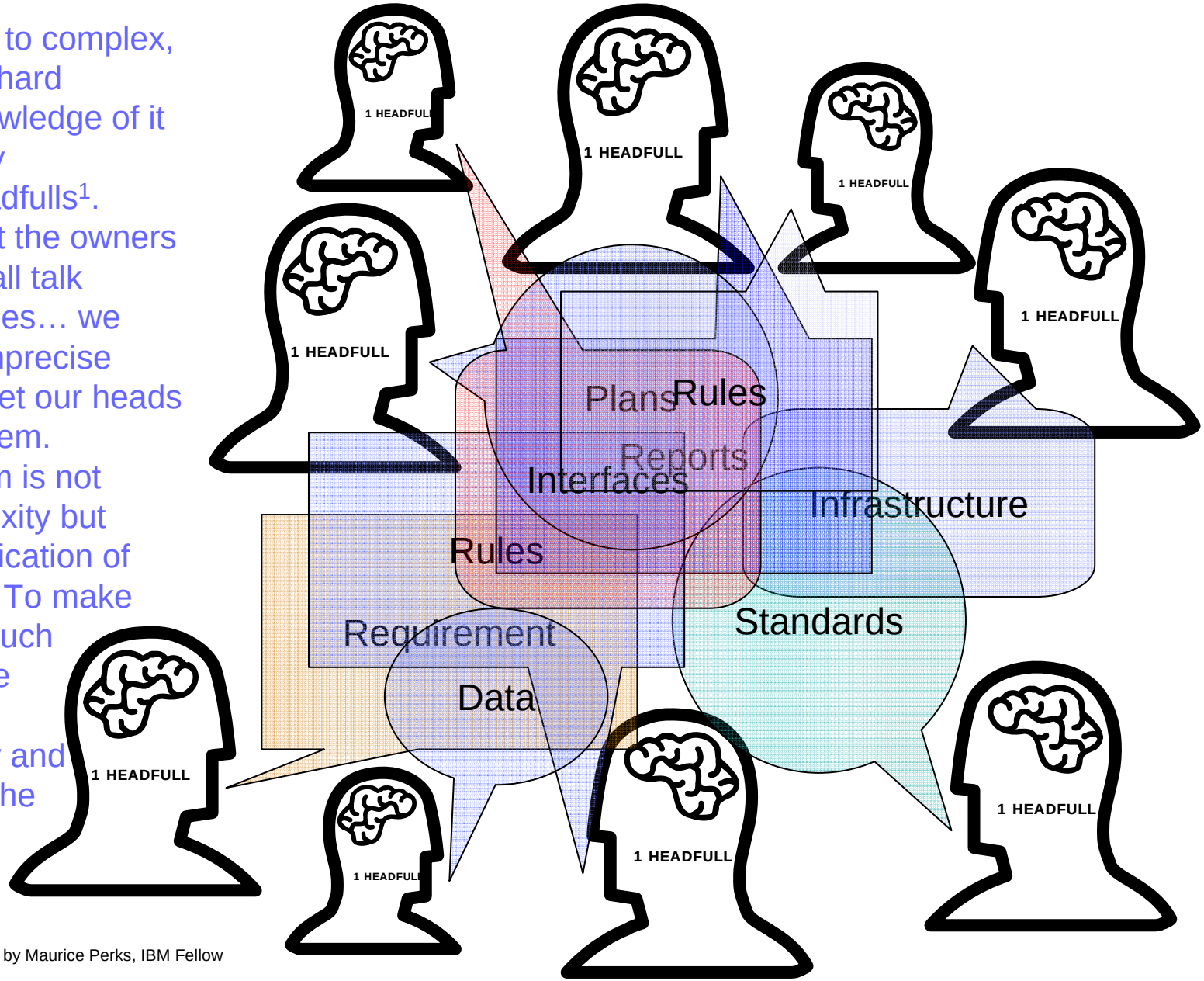


This is because of a gradual accumulation of IT complexity over the years. This makes the businesses systems more expensive to maintain, but also more expensive to integrate with or to transition from. Complex IT landscapes are like Brownfield construction sites; they are more expensive far riskier to develop on, but they are the foundation of our modern society.



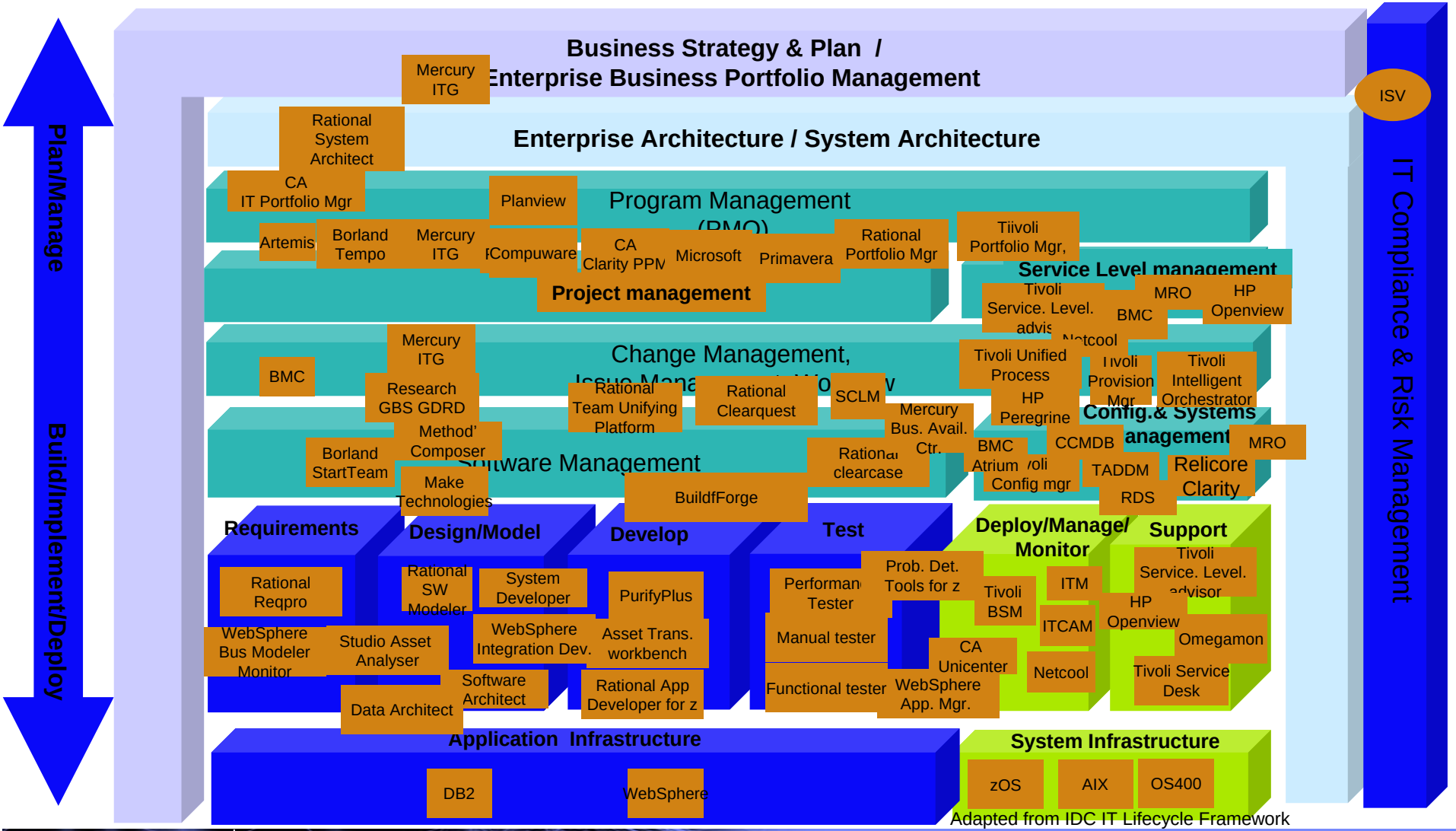
Sources: Capers Jones, *SW Repair and Rennovation in 21st Century*; Forrester *IT Spending Benchmark, Brownfield Study Findings*

Making changes to complex, intertwined IT is hard because the knowledge of it is stored in many independent headfulls[1]. Not only that, but the owners of the headfulls all talk different languages… we also often use imprecise abstractions to get our heads around the problem. Our core problem is not technical complexity but precise communication of that complexity. To make matters worse much of this knowledge is sometimes forgotten entirely and is hidden within the systems themselves…



[1] – Headfulls were invented by Maurice Perks, IBM Fellow

Just as the headfulls are split across multiple people, the tools and skills to manage this complexity are similarly fragmented, making it difficult to manage and make change….



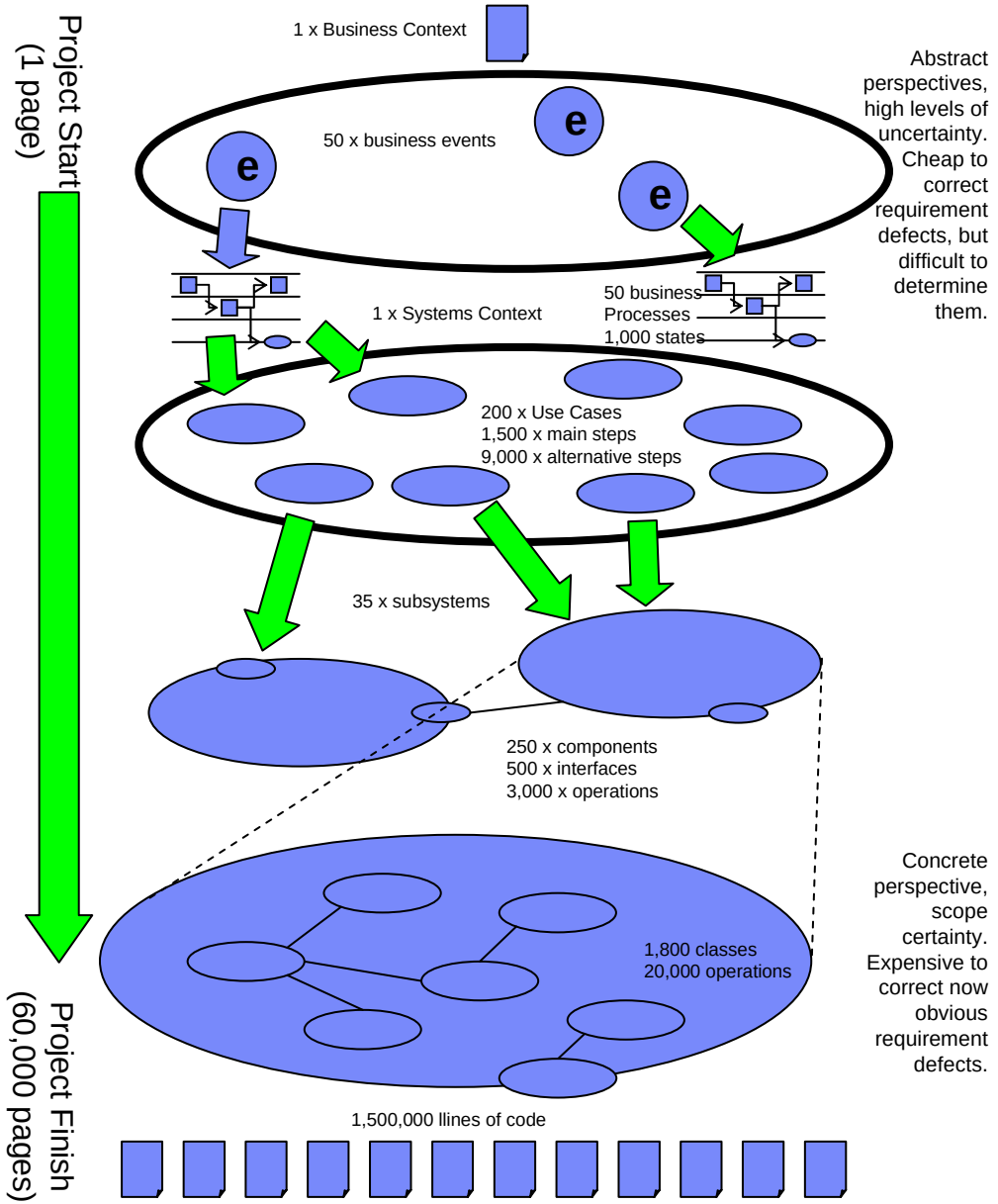
Adapted from IDC IT Lifecycle Framework

Today's methods are not in a position to address these deficiencies without augmentation. This is because most methods were evolved from academia or from large Greenfield projects.

Virtually all current methods start with "Understand the Landscape", but in reality this is unachievable and unaffordable due to IT complexity and human limitations.

As a result we begin with extremely abstract views that are decomposed over time. The mistakes and simplifications in the abstract views (at the top of the diagram) are amplified down through the lifecycle. In the later stages they are far more expensive to correct

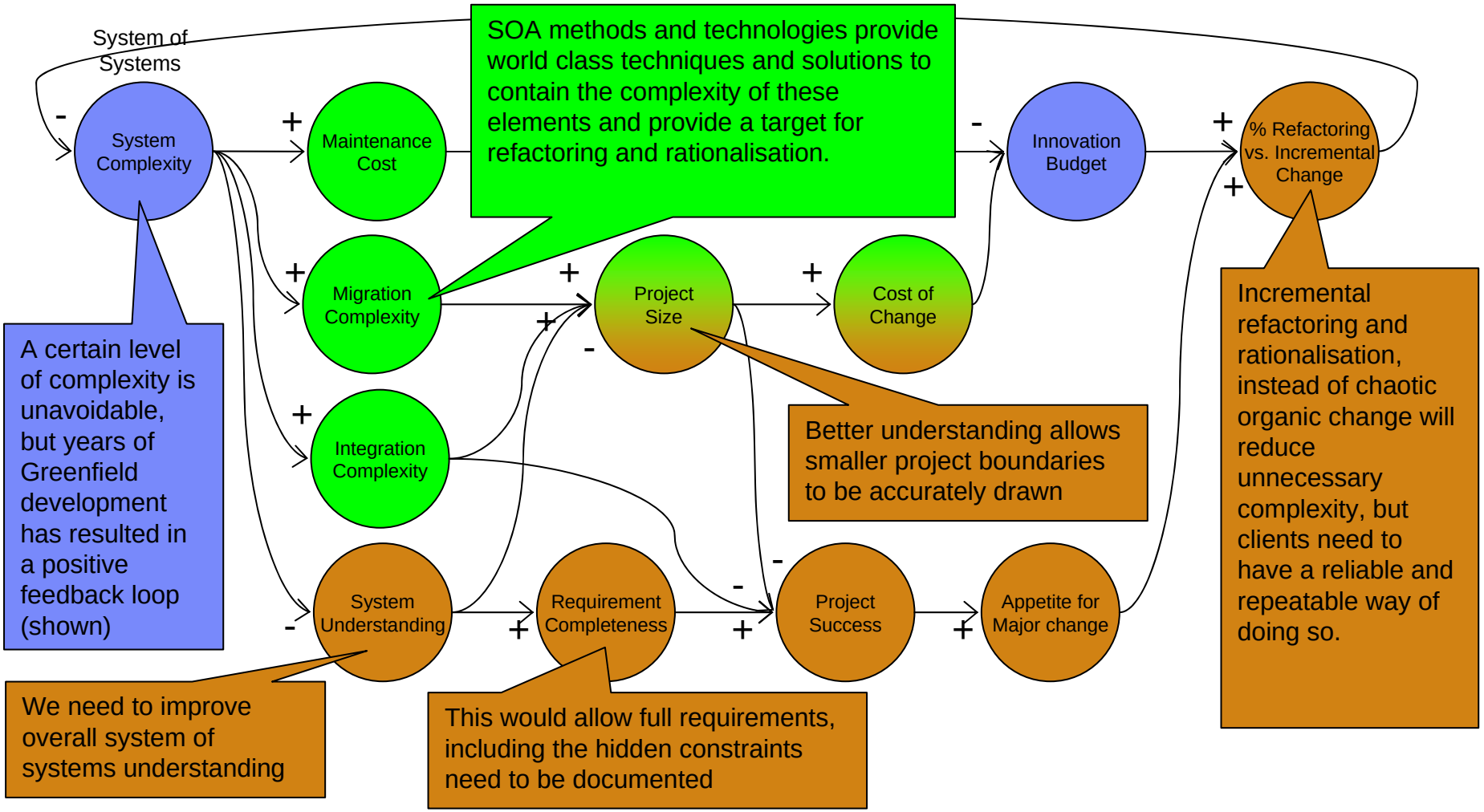
As the analysis and design defects are exposed during projects, slippages and cost overruns are inevitable as often the defects lay undetected until late stage testing…



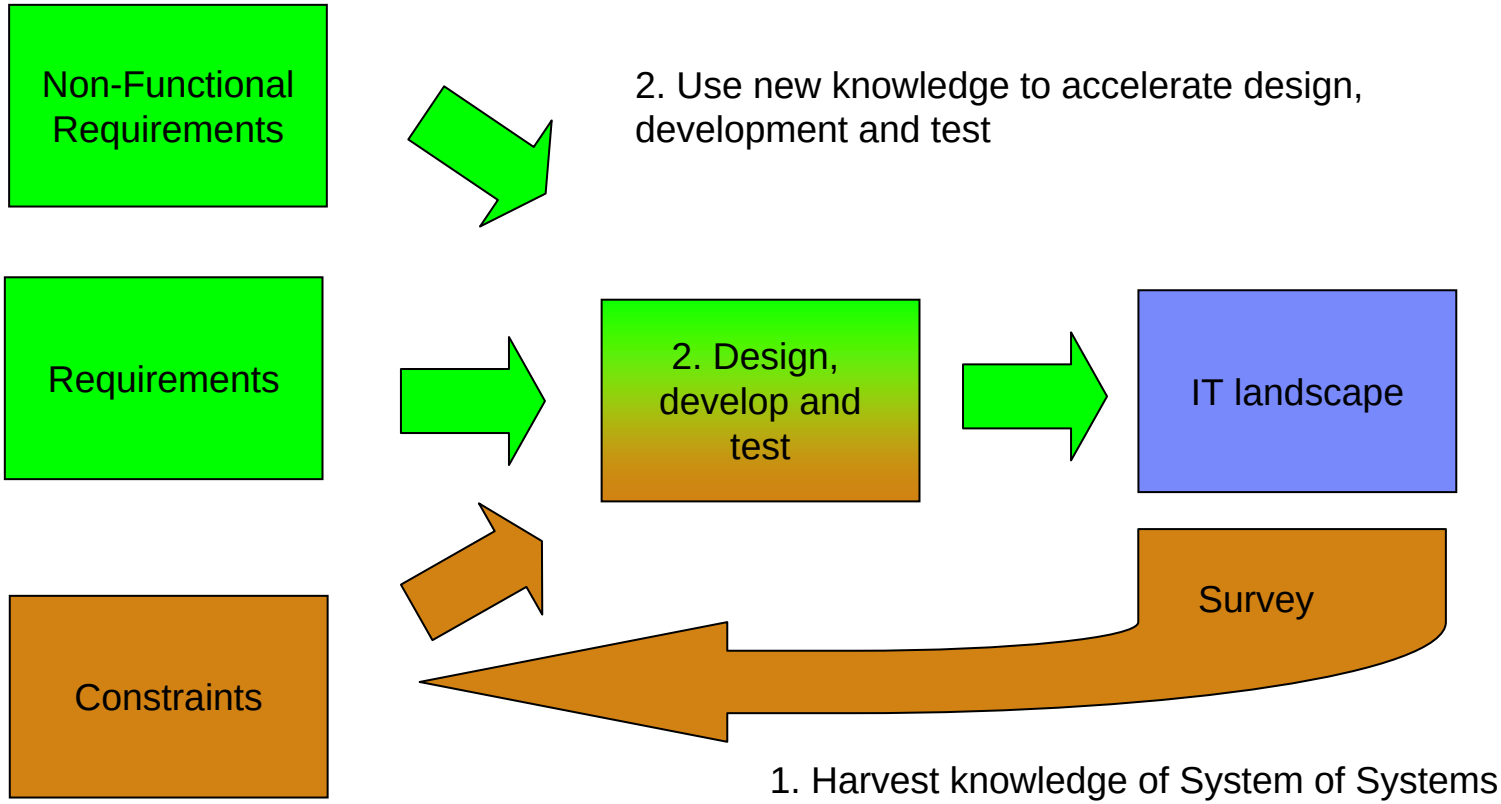
Why not start from scratch?

- This can only be done with a Greenfield business model; some businesses have created Greenfield e-businesses, but cannot ‘reboot’ their core business in the same way. This is because:
  - There is a huge amount of accumulated, encoded corporate knowledge both in systems and in data; this cannot be replaced in one go as it is too large
  - This accumulated knowledge is both intrinsic and implicit, it is rarely documented or explicit
  - Many systems and processes are unknown and are effectively ‘black box’ (especially in cases of extreme system age or extensive package use)
  - Replacing one element at a time means it must remain consistent with everything else. In this approach testing becomes major burden

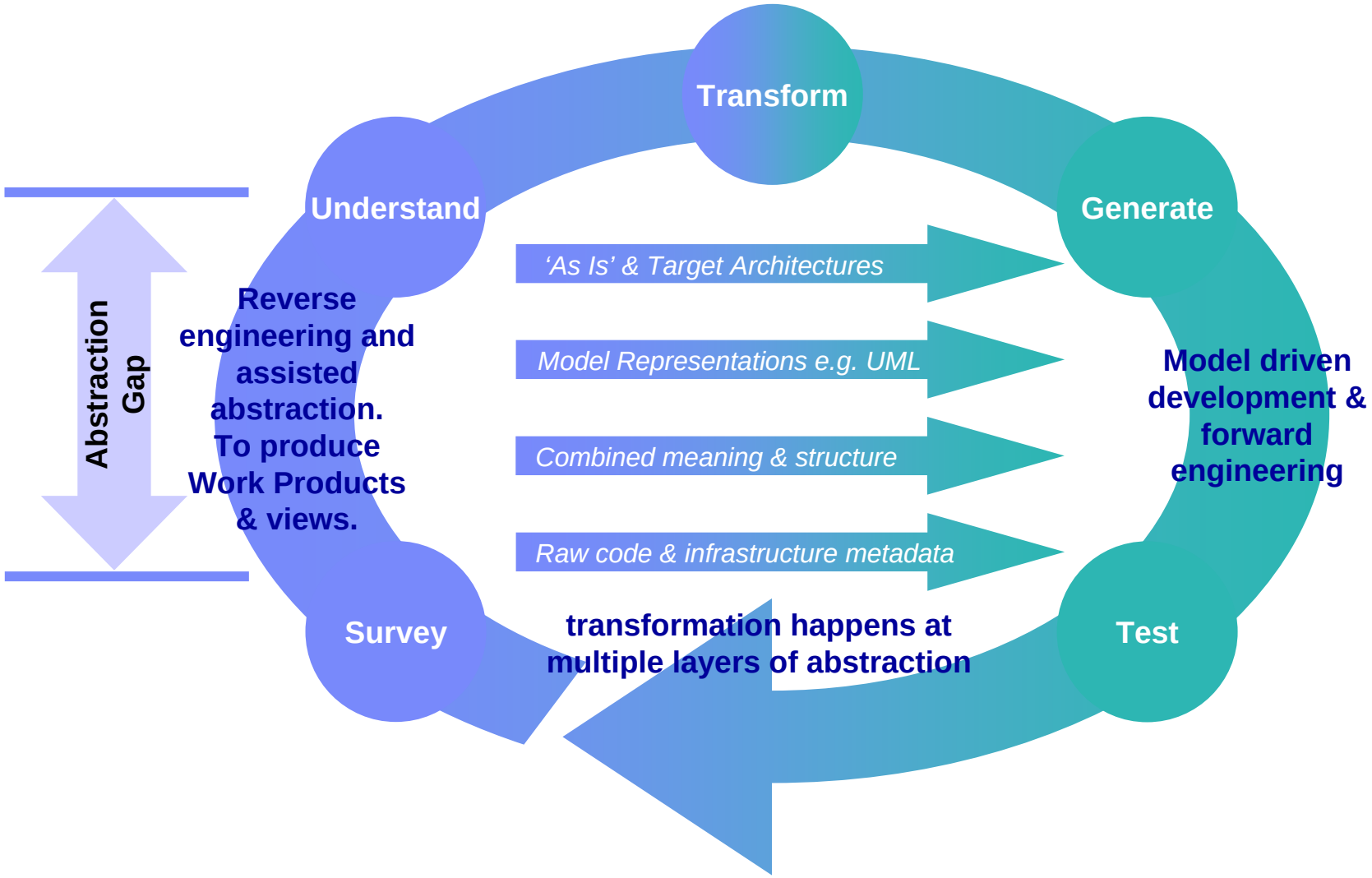
To get us out of the complexity trap, we need to augment our methods and tools to counter the feedback loop. Some approaches such as SOA provide ways to contain maintenance, migration and integration costs, but this must be amplified via a better understanding of complex systems of systems which we can only achieve through Brownfield Re-engineering tools and techniques.



To achieve this better understanding, we will need to automatically harvest this information from the IT landscape. To do so manually would be prohibitively expensive. Today's best practices of Model Driven Development and Pattern Driven Engineering would suggest that this surveyed information should then be used to generate the next version of the landscape. This is shown conceptually below:



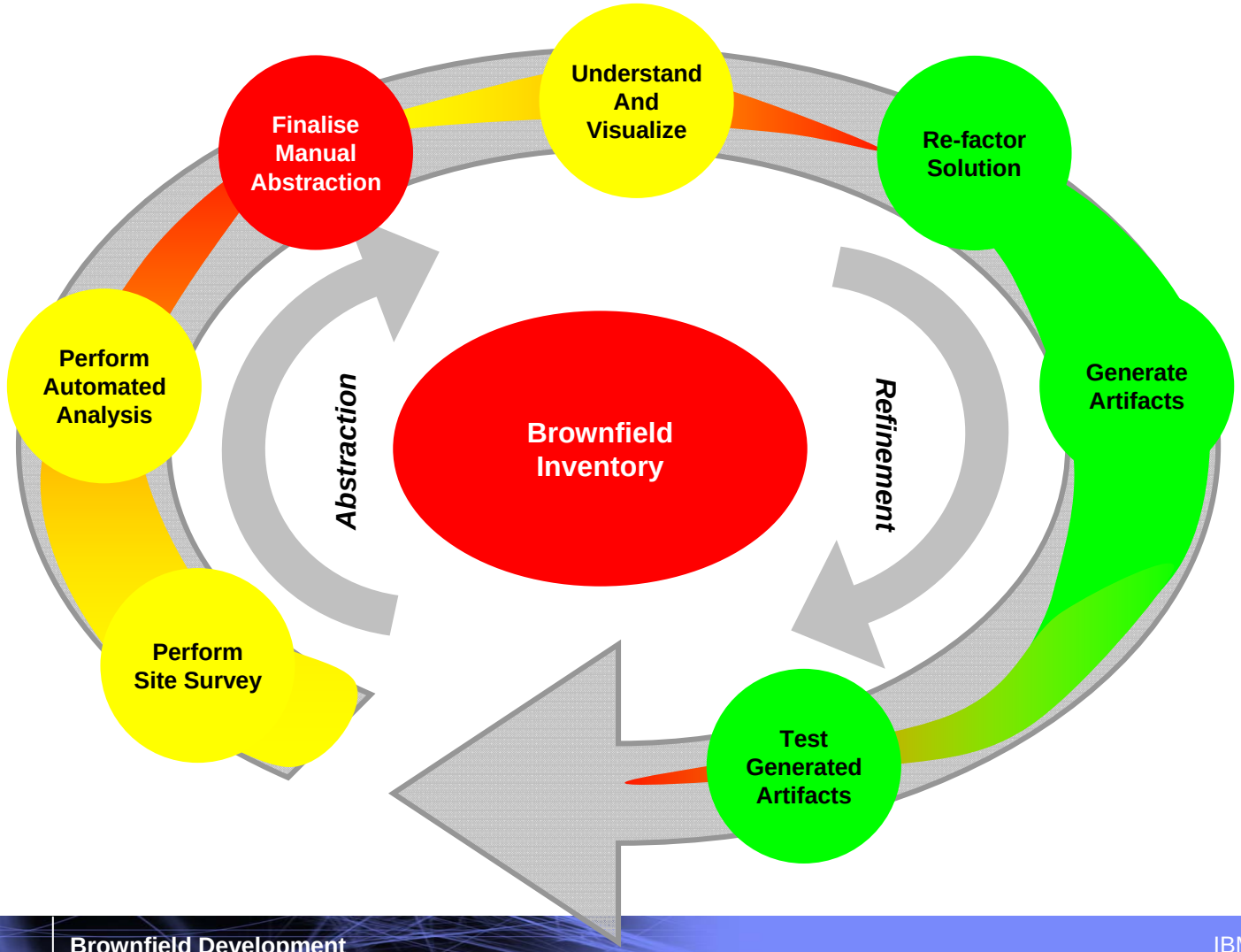
We have developed a fundamentally new Brownfield Development lifecycle which is specifically designed to absorb and re-engineer interlinked systems with high levels of complexity.



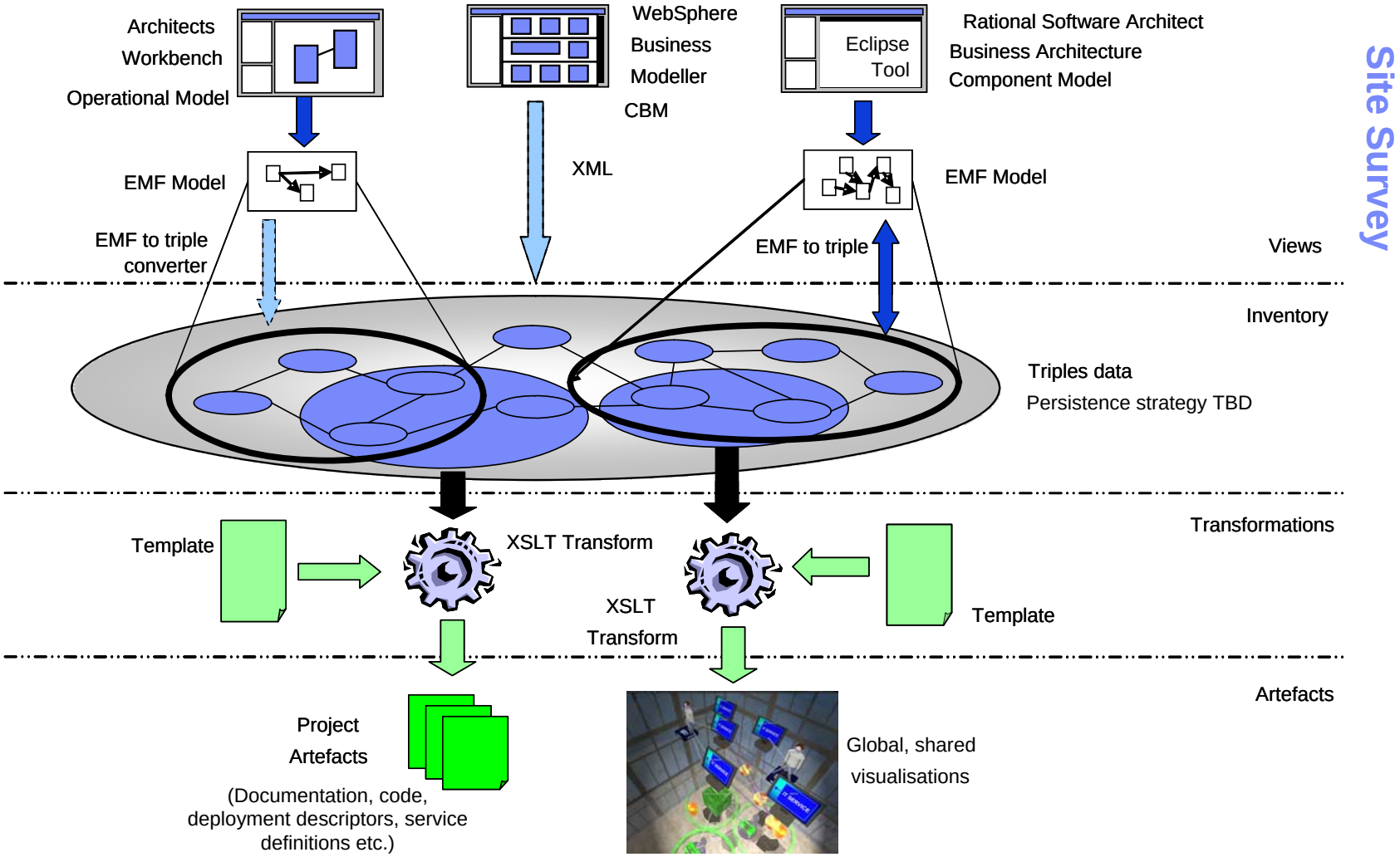
The findings of the study revealed no full support for the full lifecycle.

- Abstraction Gap
- No single tool is available and the team have concluded that there can't be due to complexity and user experience issues
  - Almost all tools are point solutions
  - Little or no integration between individual tools
  - Few examples of even partial lifecycle
  - Lots of partial point solutions, so no need to start from scratch
  - Multiple non-bridged divides (business; application; data; infrastructure)
- Integration/metamodels will be key
- Encoding and injection of industry assets will be necessary
- There are some current signs of convergence, but only via informal means (at least partially due to Study)
- Some interest in Industry but no broad agreement on anything except the fact that this is a major problem/opportunity space

We conducted a Survey of the state of the industry around the lifecycle. Each of the circles below will hyperlink to the relevant section of their report. The traffic lighting shows areas of high and low industry maturity. Thickness of lifecycle arrow shows degrees of compatibility between adjacent solutions.

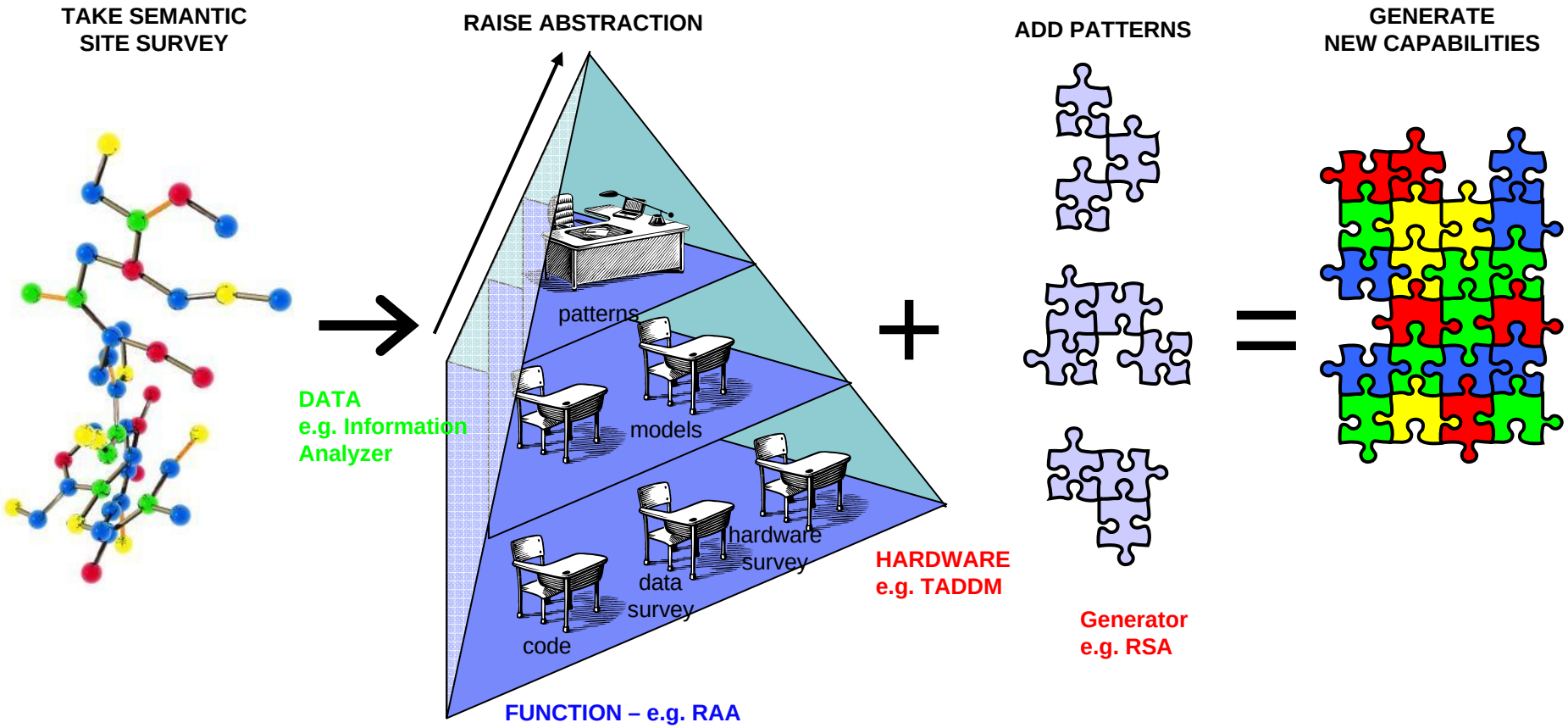


The Study has established that multiple tools need to feed into a single logical Inventory. Federation of the Inventory is probably necessary. The metamodel of the Inventory will need to be flexible, but more standards in this area will increase productivity and reuse.



Site Survey

The Brownfield lifecycle can be supported through the use of standard model driven and pattern driven engineering supplemented with the technologies of the Semantic Web to combine and abstract the information from the Surveys. Investment is required to make these tools more integrated and easier to use.



1. Capture formal Views of existing assets via Site Survey using existing tools and convert into triples. Cross reference this information and store in the Inventory.	2. Raise abstraction level of this information via automated (not automatic) aggregation, abstraction and pattern recognition.	3. Use all layers of Inventory to extract data, combine it with existing patterns and re-generate new capabilities which fit within the existing landscape.

- IBM has significant assets and tools that can and are beginning to be used to accelerate and deploy Brownfield Development capabilities including:
  - Survey tools covering infrastructure, data and application (both Software products and Consulting assets)
  - System Description Standard (Metamodel) and Rational (Telelogic) System Architect to raise levels of abstraction
  - Model Driven Development and Architecture tools and techniques to forward engineer
- Brownfield Development is described in more detail in the book *Eating the IT Elephant* published by IBM Press in 2008


Richard Hopkins


Kevin Jenkins


Grady Booch


Chris Winter

Supporting Material

Bibliography

- Brownfield – http://en.wikipedia.org/w/index.php?title=Brownfield_%28software_development%29&action=history
- Component http://en.wikipedia.org/wiki/Component_%28UML%29
- Operational Modelling - http://publib.boulder.ibm.com/infocenter/rsawshlp/v7r5m0/index.jsp?topic=%2Fcom.ibm.ccl.soa.deploy.core.doc%2Ftopics%2Fstructure_logical_tsk.html
- UML - http://en.wikipedia.org/wiki/Unified_Modeling_Language