



Corporate Glue

Network-based systems integration

Richard M Marshall
CTO, Prosumer Solutions Ltd

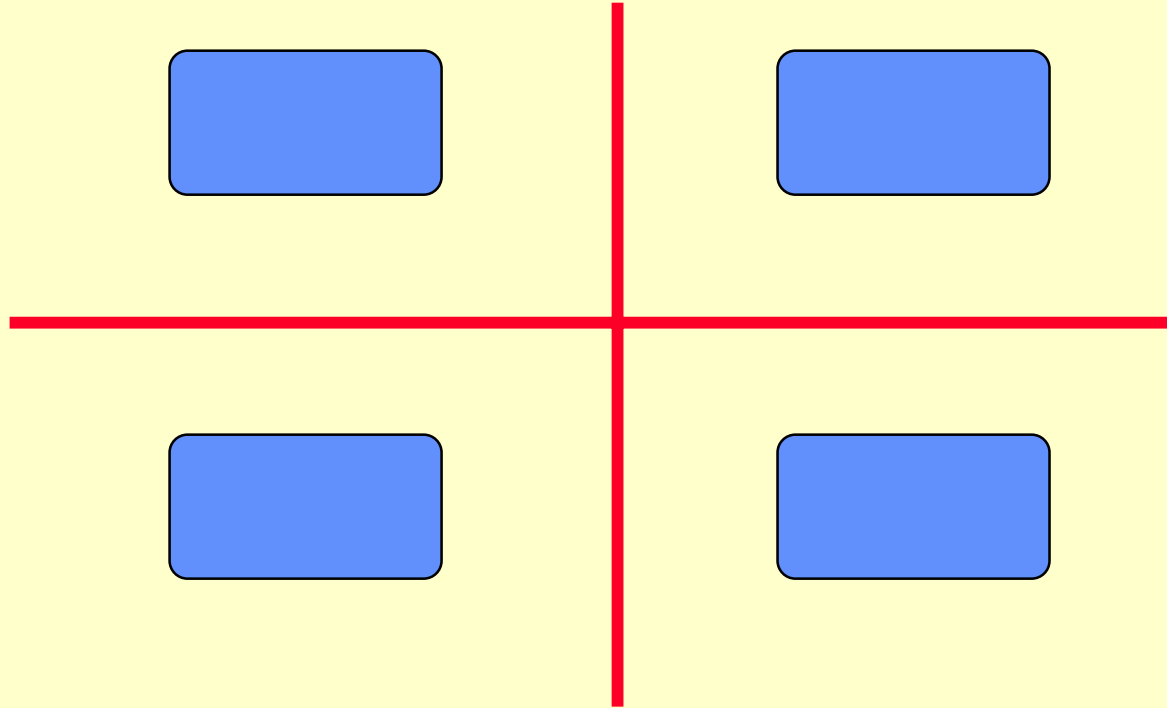


Contents

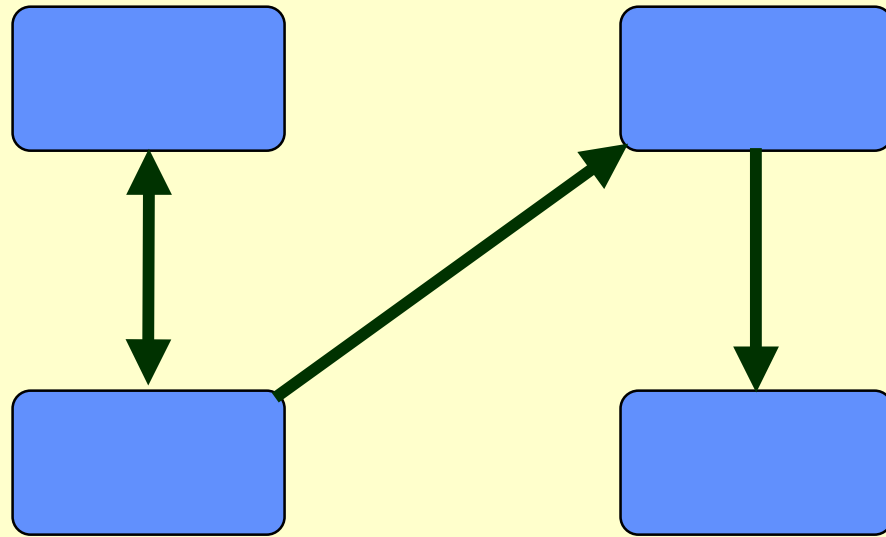
- Systems evolution
- Network enabling technologies
- The what and the how
- Companies, networks and communications
- Network lifecycles
- Making it happen

- Interspersed with discussion points

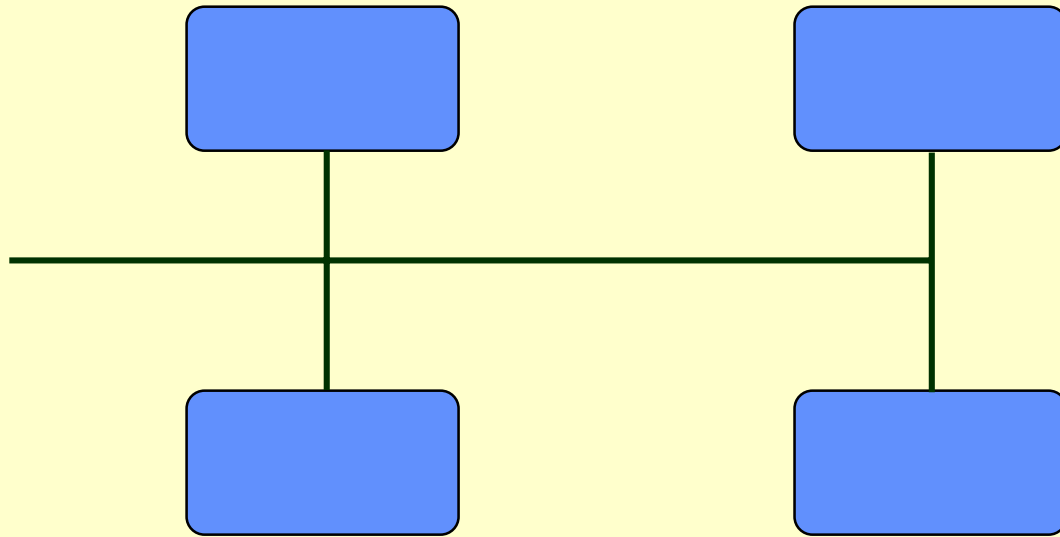
Splendid isolation



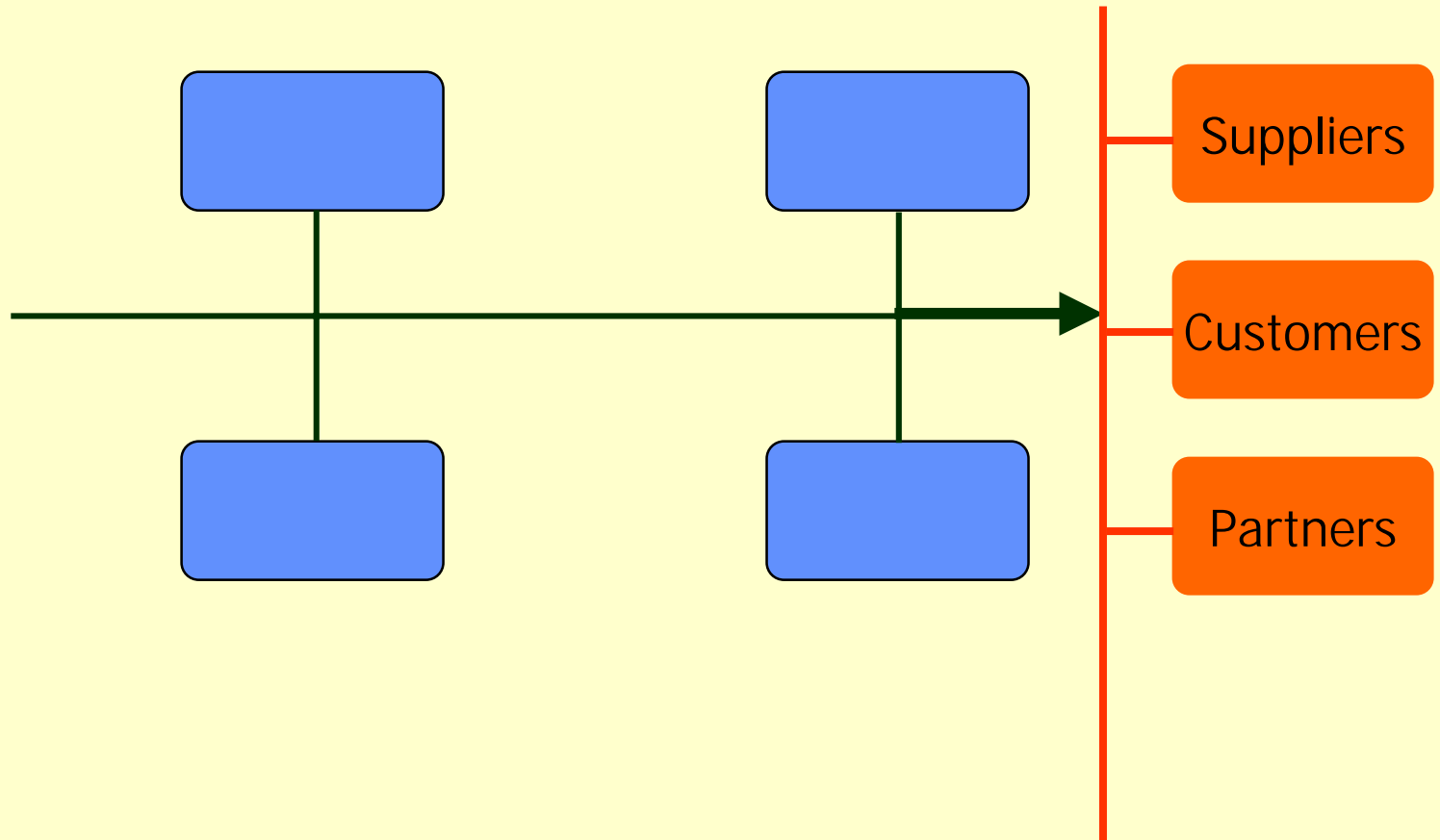
Point-to-point integration



Network integration



Beyond the enterprise



Systems approach

- Point to point communication
 - Even if running over a general network
- Application specific
- Proprietary
- Comparatively cheap to set up
- Costly to maintain
- Expedient point solution
- Tactical approach

Network approach

- Based on ubiquitous communications
- Generic solutions
- Standards based
- Expensive to engineer well
 - But open to cheap approaches
- Lower cost to maintain
- Strategic approach
- Must be part of long-term commitment

Enabling technologies

- Public, standardized Internet protocols
 - HTTP and TCP/IP
 - XML
 - SOAP, Microsoft .net
- More to component- and server-based development
- Not intrinsically better, just everywhere
- Generally simple
- Vendor independent (well, more or less)

XML

- eXtensible Markup Language
 - A relation of SGML and HTML
- There is a huge amount of hype around XML
- XML is a “non-standard standard”
- XML is nothing without:
 - A DTD or schema
 - Code that understands the data’s semantics
- However:
 - There are many good tools - often free
 - Text-based, so easily debugged and implemented
 - Vendor independent
 - Object friendly

SOAP

- Simple Object Application Protocol
- Allows programs to interact across the network
- A modern implementation of RPC
 - Built on HTTP and XML
 - Wide industry support
 - Text-based approach avoids data conversion problems
 - The basis for Microsoft .NET architecture

Components and servers

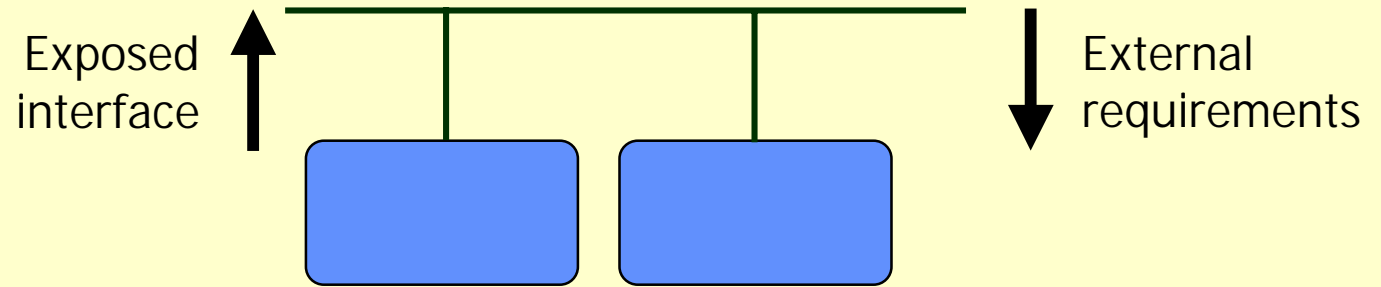
- Vast range of industrial-strength components
 - ComponentOne, Data Dynamics, IBM, Infragistics, Microsoft and many others for Windows development
- Increasing range of ready-to-run server applications
 - IBM, Microsoft, Oracle and many others
- Development becomes gluing these together
 - Understanding the business context over coding
 - Less fundamental programming
 - Less scope for core bug insertion
- These make it easier to adopt standards
 - No need to code up your interface libraries
 - Guaranteed interoperability

So what's the problem?

The “**how**” of networking isn't
the problem any more

We need the
“**what**”
and the
“**why**”

Meaningful connections



What makes sense?

The publisher's dilemma

- I have to expose an interface
- How do I know what to offer?
- How do I know who is using the service?

Discussion:

- Can we find a systems-engineering approach?
- Are there any similar situations in classic SE?

Communications

- Pervasive networking drives human communication
 - Devices can't understand each other without us
 - Semantics can't be derived from formats
- Systems interaction requires team interaction
 - Technical team to technical team
 - Business team to business team

Corporate and network structures

- Integrated network requires an integrated company
 - Convergence of systems and the corporation
 - Beware of consultant speak here!
~~“Integrated enterprise ecosystems integrating LOB-driven information silos creating cooperating empowered constituencies.”~~
- Simple, planned evolution
 - Buzzword free implementation plan
 - Buy in from key parties
 - Educate and involve
- Global reach
 - Requires board-level approval for full benefit

Does this affect you?

Discussion:

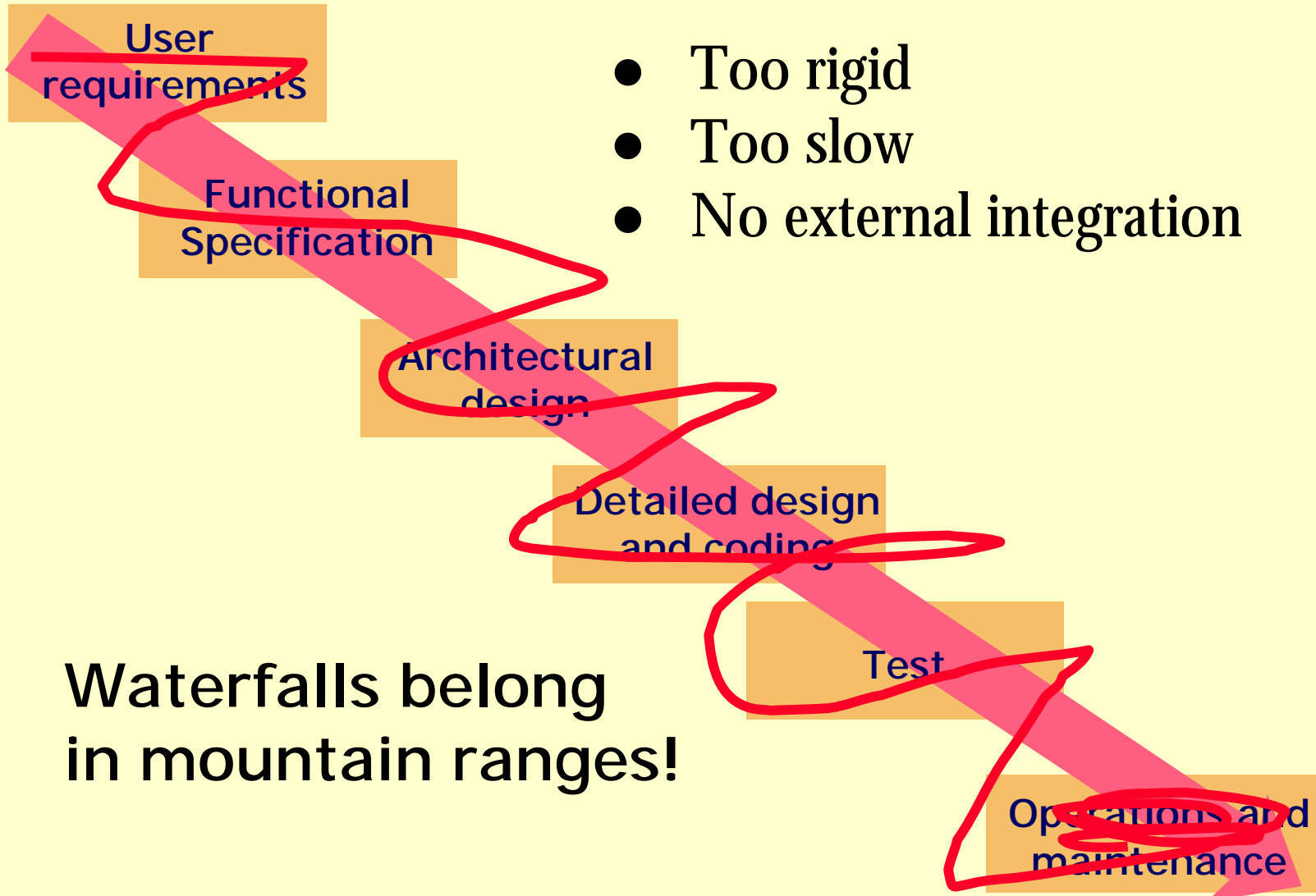
- Are you seeing network and corporate convergence?
- Do you expect it?
- How is/would implementation happen?
- Does it apply world wide?
- Is it making life easier, or just creating new problems?

The network lifecycle

What is the impact on project lifecycles?

- Formative phase is even more key
- Implementation is heavily driven by available tools
- Systems no-longer have independent lifecycles
- Controlled evolution is essential:
 - Flexibility, *yet*
 - Stability

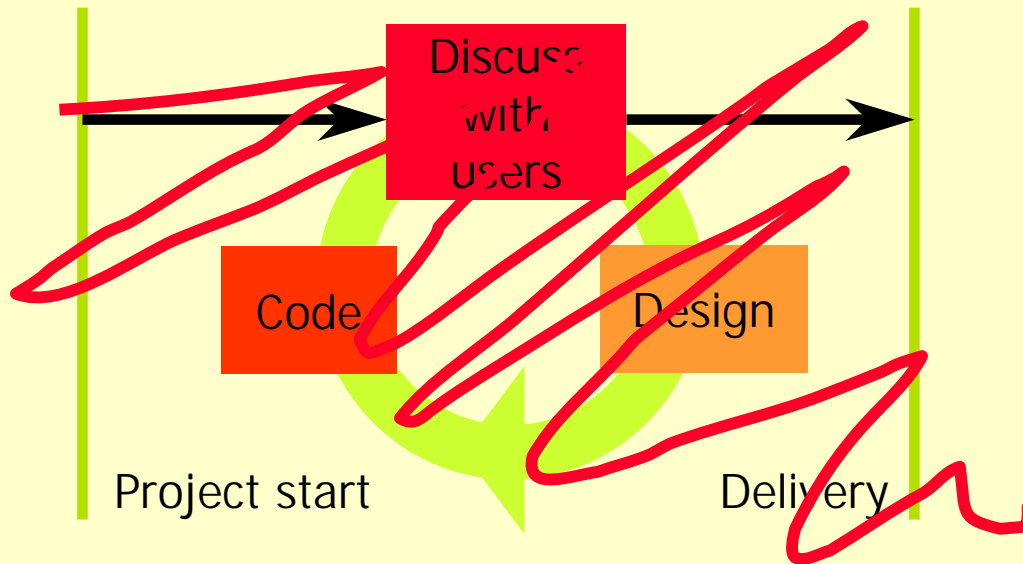
Classic waterfall model



- Too rigid
- Too slow
- No external integration

Waterfalls belong
in mountain ranges!

RAD



**Never
stabilizes!**

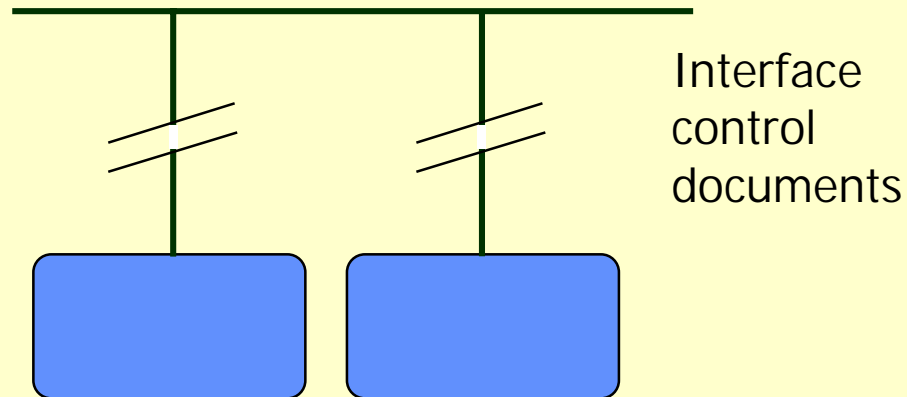
- **RAD advantages:**
 - Fast
 - User-driven
 - Point solutions
 - Suits rapidly evolving needs
- **RAD problems:**
 - Maintenance nightmare
 - No documentation
 - No business integration
 - Difficult to plan and manage

eXtreme Programming



- Don't be put off by the name
- A fundamentally sensible approach, based on practical experience
- Intended for smaller software projects
- Main principles:
 - Focus on business needs
 - Iterations of task-based planning
 - Pair programming
 - Communal ownership of code
 - Write tests first and test continuously
 - Only do what is needed today
 - Refactor mercilessly

The jigsaw lifecycle



- Each integrated system defines an ICD
- Prepared in collaboration with the other systems
- Strong version control required
- The internal lifecycle is less important, provided:
 - The interface is respected
 - Planning control points for coordination are met

Network ICD

- ICDs may be classic SE, but unusual in software
- Currently interfaces are defined technically
 - IDL to RPC
 - WSDL (Web Services Description Language) for SOAP
 - Defines parameter data types, not semantics
- Version control is very important
 - WSDL controls supported versions
 - Applications can check availability of desired version, *but*
 - The desired version must be available

Is this enough?

Discussion:

- Classic ICDs are fairly static, will these change?
- How can we define an ICD for a new system?
- What else can systems engineering offer here?



Making it happen

- Systems engineering is only valuable when applied
- All too easy to make the right noises but do nothing
- Politics will get in the way
 - Intersystem integration implies corporate politics
 - Need board-level buy-in
 - This is not IT driven – it must be business driven
- Technology can help make it happen




Creative approaches to failure

- Management By Buzzword
 - Dogma “it works fine for manufacturing”
- Processes not adapted to project needs
 - Inflexibility “thou shalt adhere to the standard”
- The benefits are not explained
 - Imposition “you lazy programmers will do this”
- Too abstract
 - Guru “this works, you don’t need to know why”
- Incomprehensible
 - Consultant “he must be good, I didn’t understand a word”

Collaborative software

- Software exists to help people collaborate
- Structured discussion
 - Groove
 - Lotus Notes
 - Microsoft Office XP
 - “News”
 - StarTeam – integrated with source code control
 - Threaded discussion Web software
- Document control
 - DOORS etc



Moving beyond the corporation

- How do we take networking outside our company?
- Which cooperation is more difficult?
 - Technology
 - Commercial / managerial
- Traditional approach of working groups?
 - Too slow
 - Too expensive
- Expedient techniques
 - Apply standards where they exist
 - Use specialist external companies

Old or new?

Discussion:

- Networking technology is new, but are the methods?
- Do we need to change our methods?
- Do we need new methods?
- If so, why?
- How do we balance flexibility and the perceived need for speed with the real need for stabilization?
- Or is this the Emperor's New Clothes?