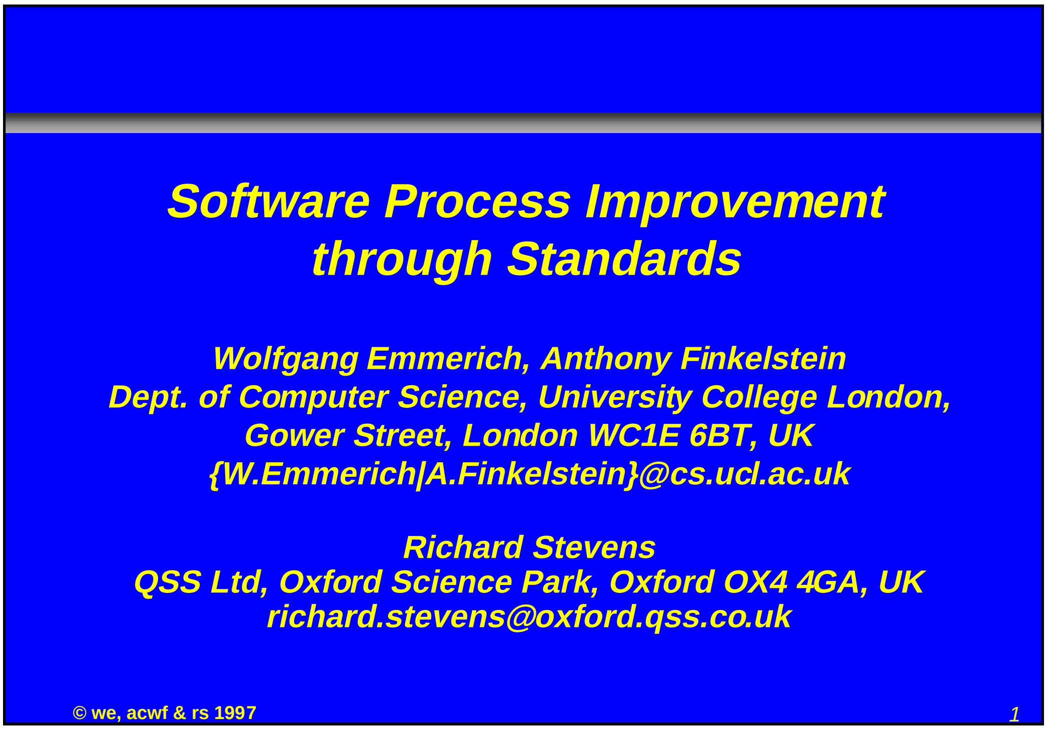
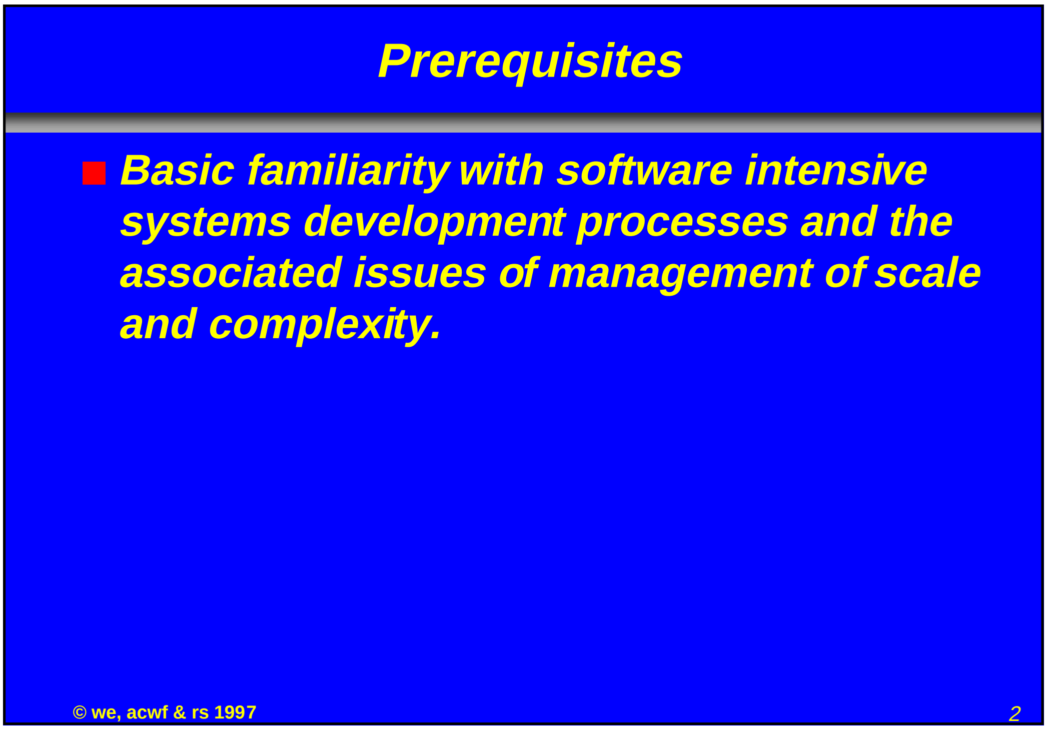


Software Process Improvement through Standards

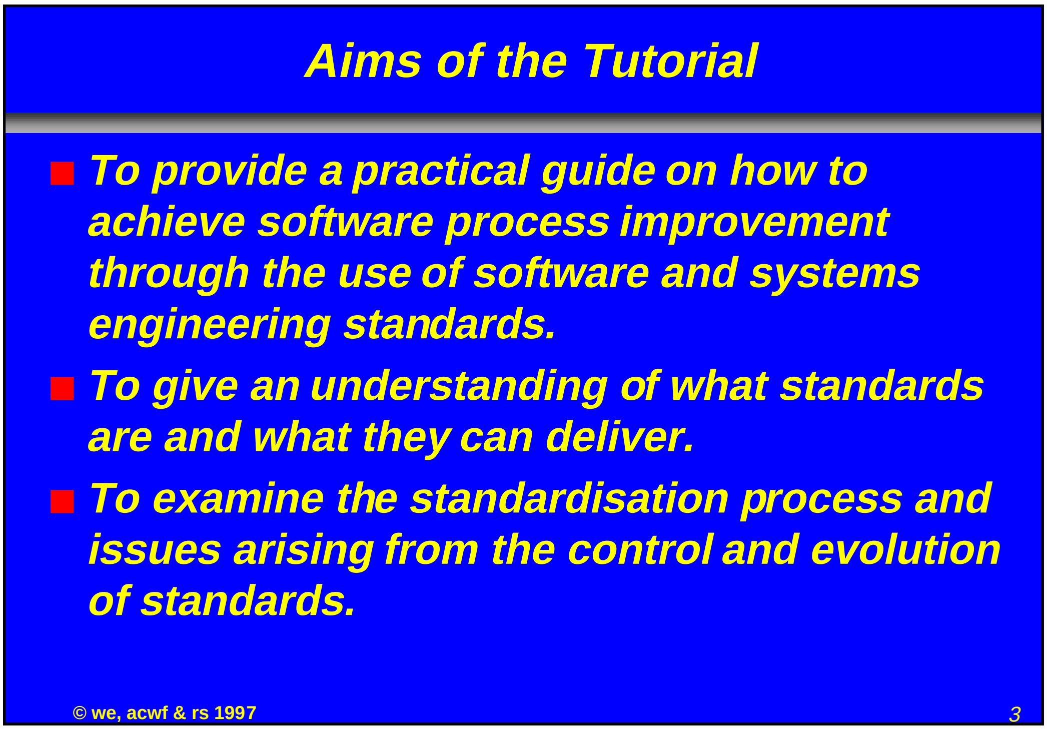
Wolfgang Emmerich, Anthony Finkelstein
Dept. of Computer Science, University College London,
Gower Street, London WC1E 6BT, UK
{W.Emmerich|A.Finkelstein}@cs.ucl.ac.uk

Richard Stevens
QSS Ltd, Oxford Science Park, Oxford OX4 4GA, UK
richard.stevens@oxford.qss.co.uk



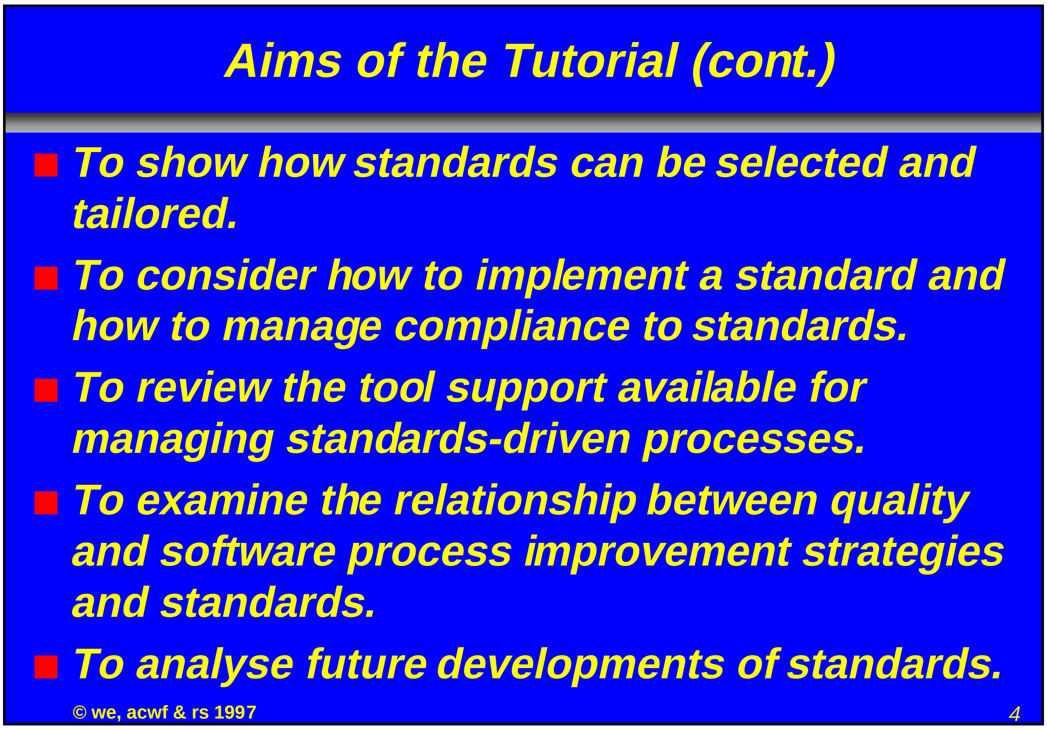
Prerequisites

- Basic familiarity with software intensive systems development processes and the associated issues of management of scale and complexity.



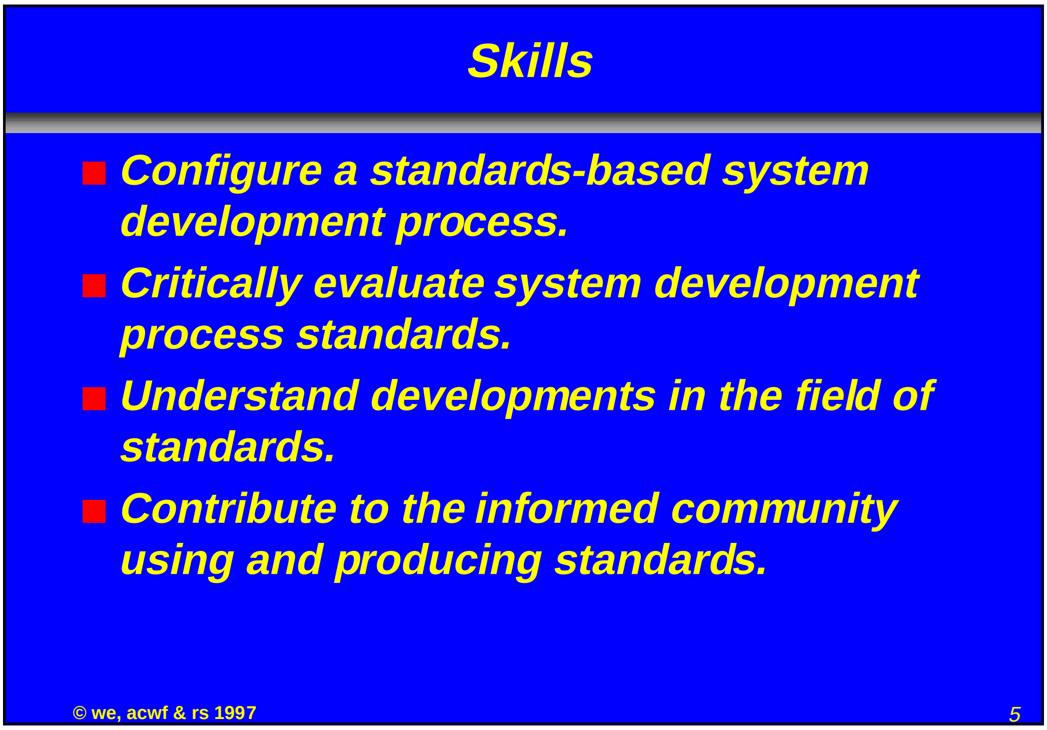
Aims of the Tutorial

- To provide a practical guide on how to achieve software process improvement through the use of software and systems engineering standards.
- To give an understanding of what standards are and what they can deliver.
- To examine the standardisation process and issues arising from the control and evolution of standards.



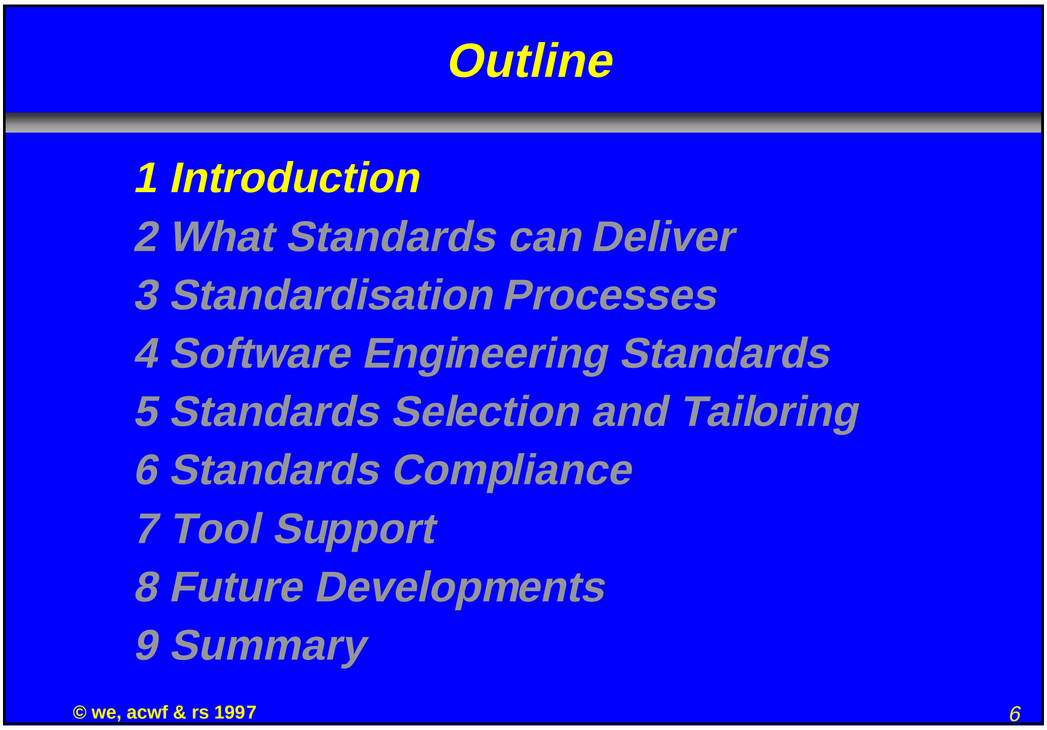
Aims of the Tutorial (cont.)

- To show how standards can be selected and tailored.
- To consider how to implement a standard and how to manage compliance to standards.
- To review the tool support available for managing standards-driven processes.
- To examine the relationship between quality and software process improvement strategies and standards.
- To analyse future developments of standards.



Skills

- Configure a standards-based system development process.
- Critically evaluate system development process standards.
- Understand developments in the field of standards.
- Contribute to the informed community using and producing standards.



Outline

1 Introduction

2 What Standards can Deliver

3 Standardisation Processes

4 Software Engineering Standards

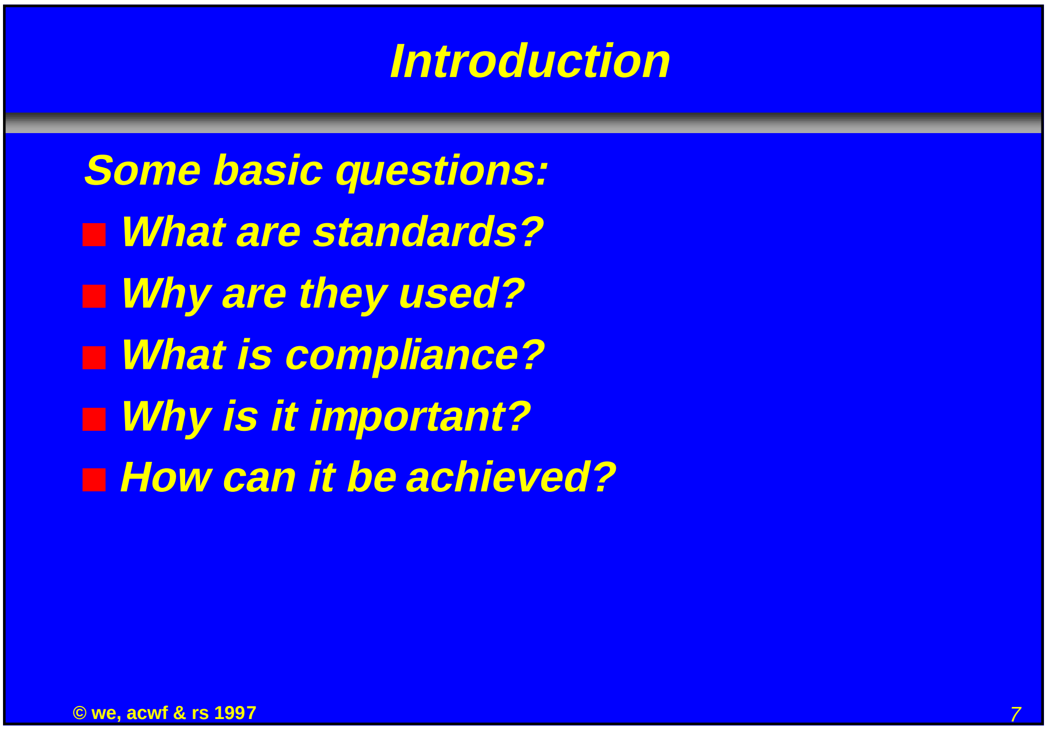
5 Standards Selection and Tailoring

6 Standards Compliance

7 Tool Support

8 Future Developments

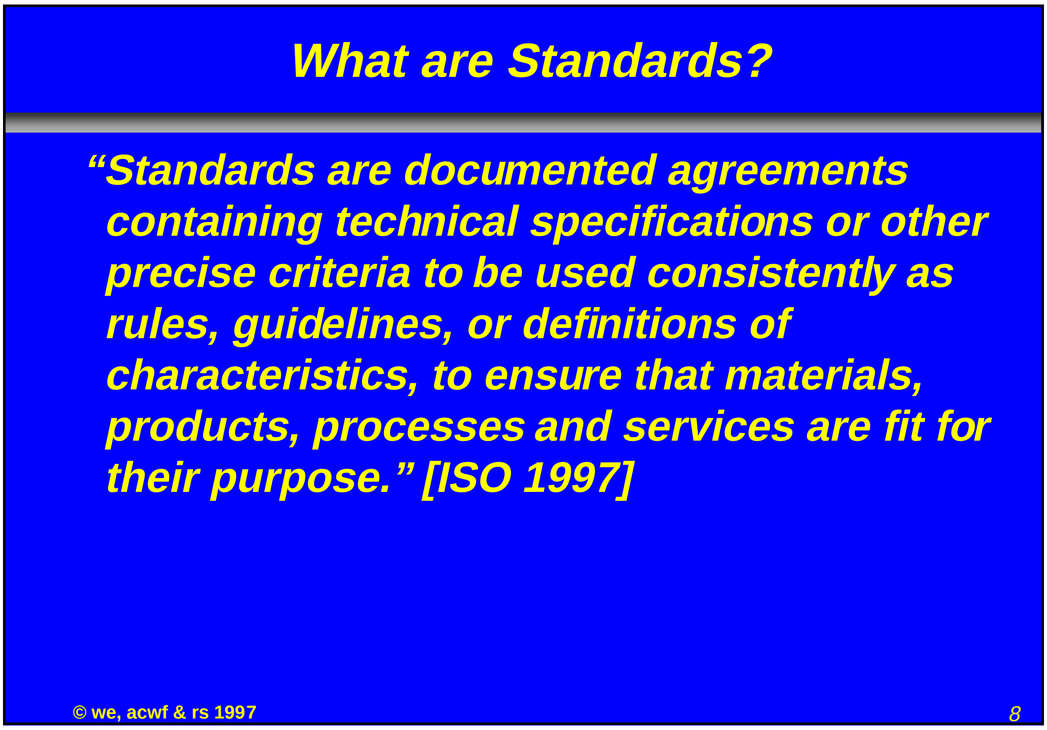
9 Summary



Introduction

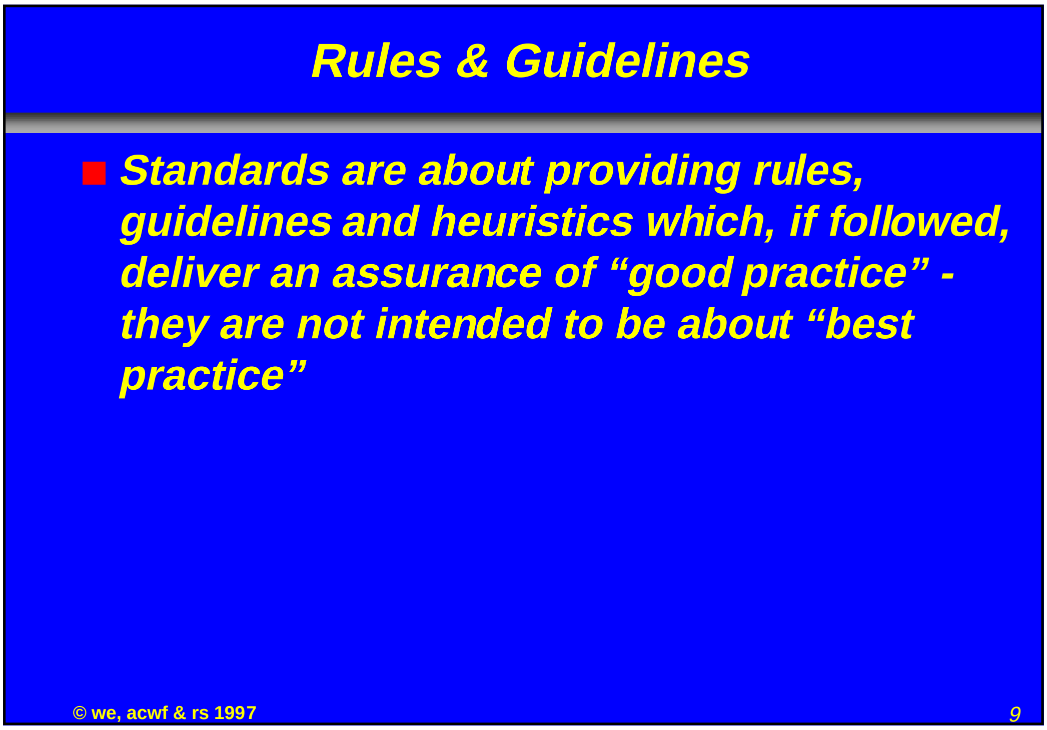
Some basic questions:

- *What are standards?*
- *Why are they used?*
- *What is compliance?*
- *Why is it important?*
- *How can it be achieved?*



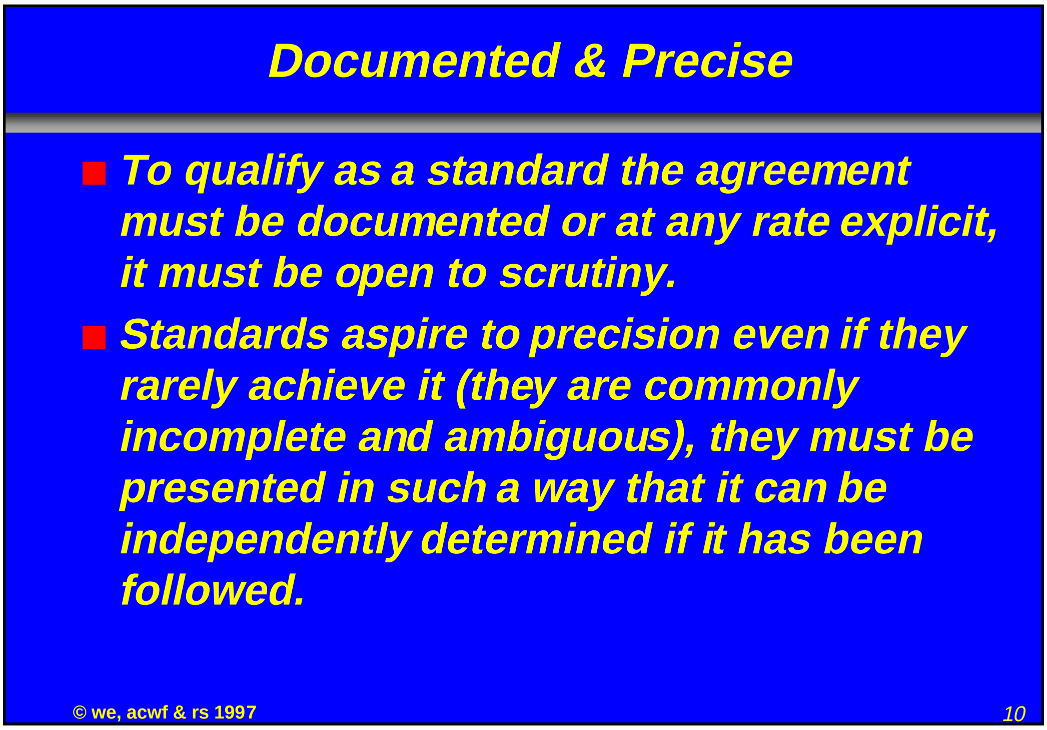
What are Standards?

“Standards are documented agreements containing technical specifications or other precise criteria to be used consistently as rules, guidelines, or definitions of characteristics, to ensure that materials, products, processes and services are fit for their purpose.” [ISO 1997]



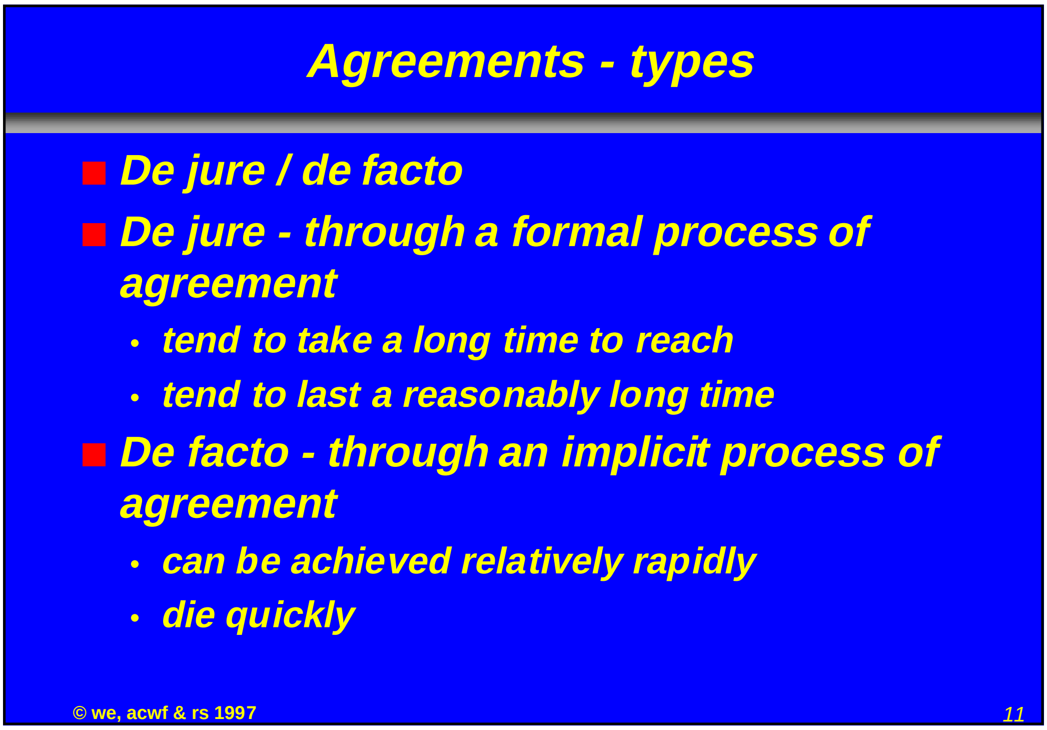
Rules & Guidelines

- *Standards are about providing rules, guidelines and heuristics which, if followed, deliver an assurance of "good practice" - they are not intended to be about "best practice"*



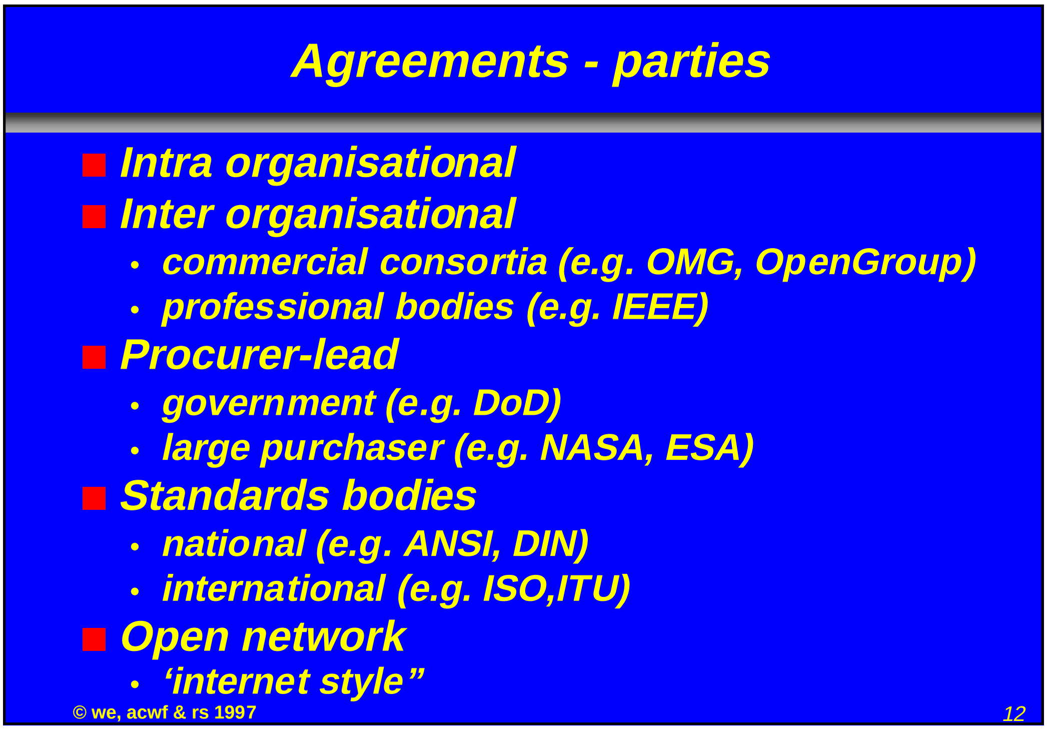
Documented & Precise

- *To qualify as a standard the agreement must be documented or at any rate explicit, it must be open to scrutiny.*
- *Standards aspire to precision even if they rarely achieve it (they are commonly incomplete and ambiguous), they must be presented in such a way that it can be independently determined if it has been followed.*



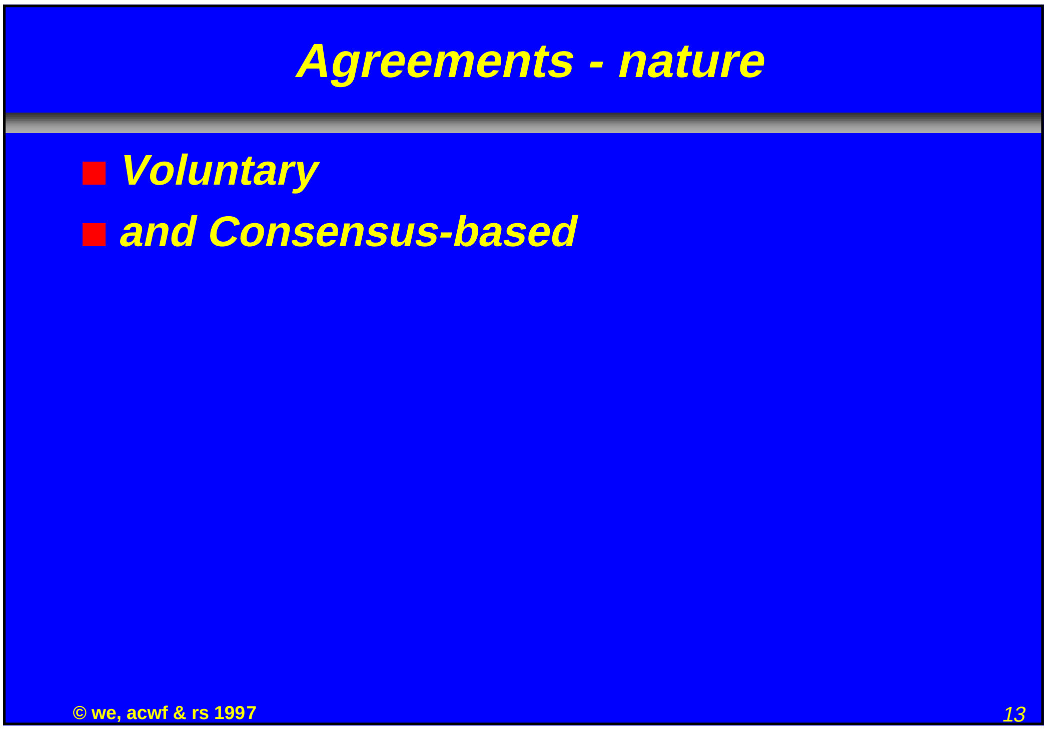
Agreements - types

- De jure / de facto
- De jure - through a formal process of agreement
  - tend to take a long time to reach
  - tend to last a reasonably long time
- De facto - through an implicit process of agreement
  - can be achieved relatively rapidly
  - die quickly



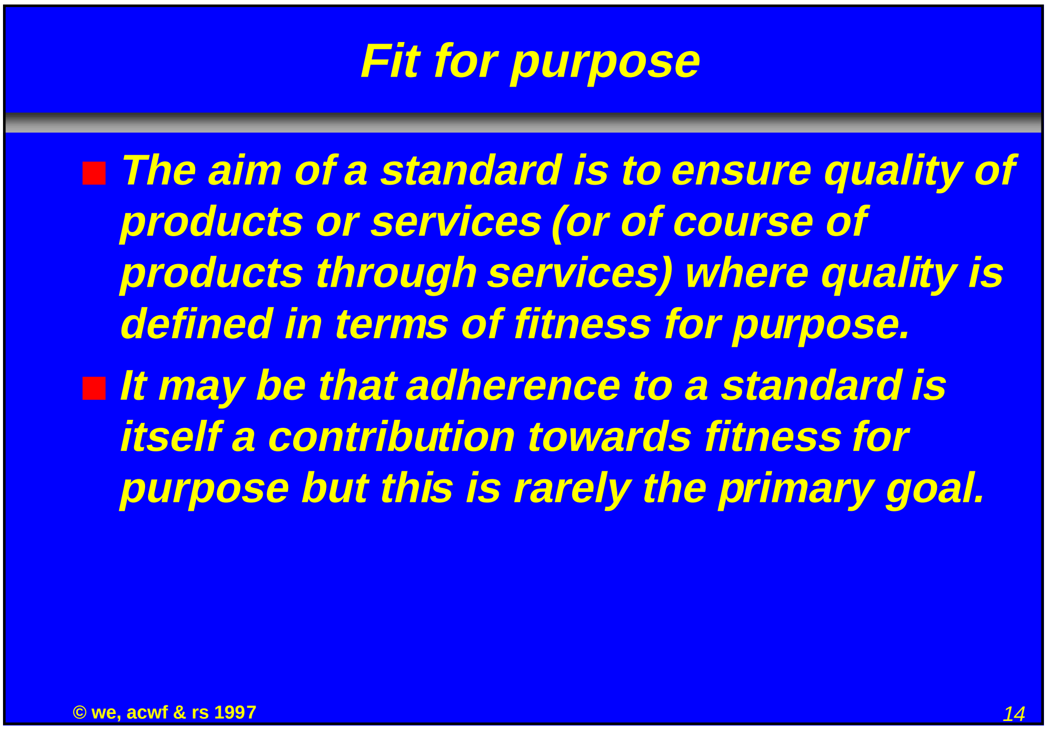
Agreements - parties

- Intra organisational
- Inter organisational
  - commercial consortia (e.g. OMG, OpenGroup)
  - professional bodies (e.g. IEEE)
- Procurer-lead
  - government (e.g. DoD)
  - large purchaser (e.g. NASA, ESA)
- Standards bodies
  - national (e.g. ANSI, DIN)
  - international (e.g. ISO,ITU)
- Open network
  - ‘internet style”



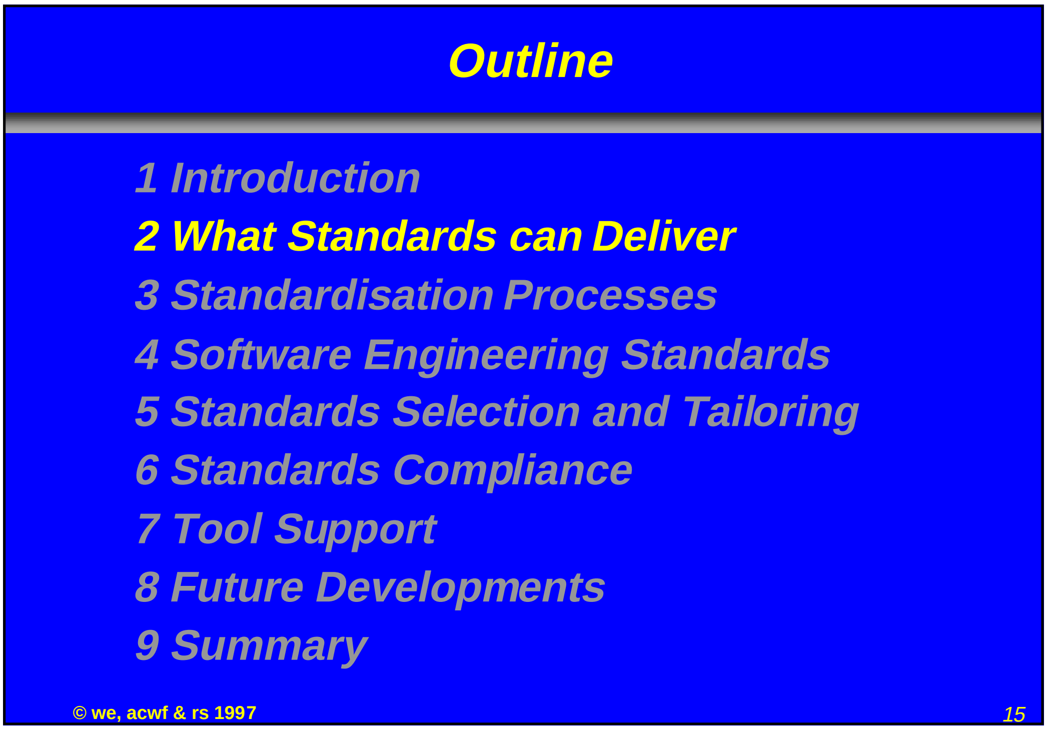
Agreements - nature

- *Voluntary*
- *and Consensus-based*



Fit for purpose

- *The aim of a standard is to ensure quality of products or services (or of course of products through services) where quality is defined in terms of fitness for purpose.*
- *It may be that adherence to a standard is itself a contribution towards fitness for purpose but this is rarely the primary goal.*



Outline

1 Introduction

2 What Standards can Deliver

3 Standardisation Processes

4 Software Engineering Standards

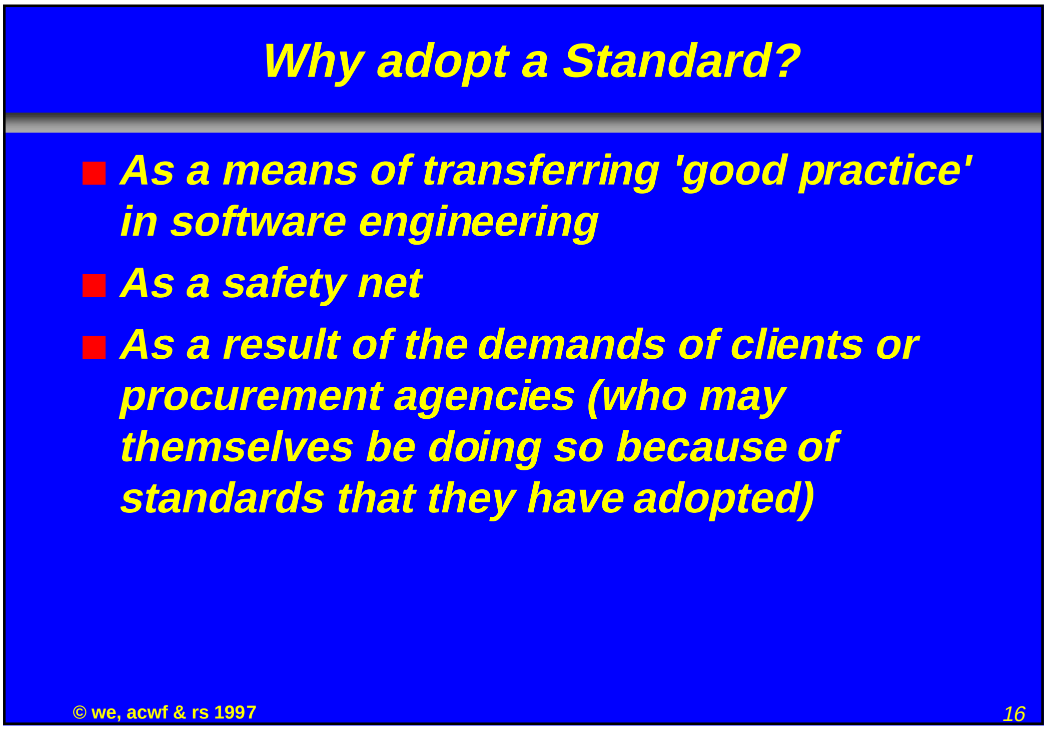
5 Standards Selection and Tailoring

6 Standards Compliance

7 Tool Support

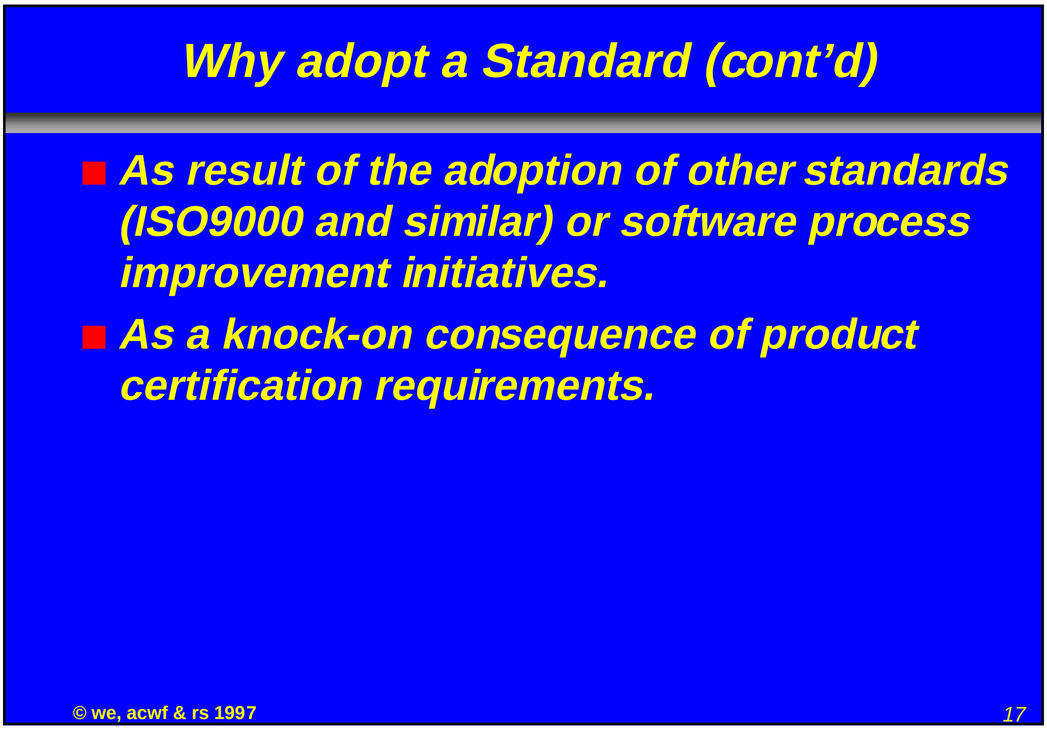
8 Future Developments

9 Summary



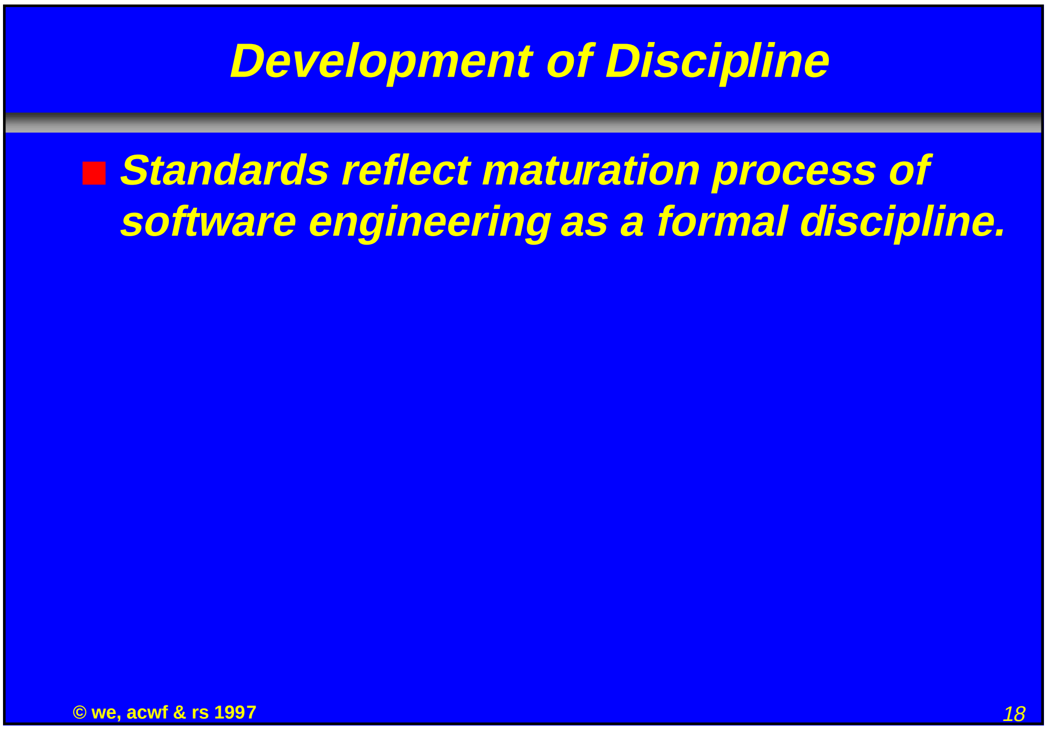
Why adopt a Standard?

- As a means of transferring 'good practice' in software engineering
- As a safety net
- As a result of the demands of clients or procurement agencies (who may themselves be doing so because of standards that they have adopted)



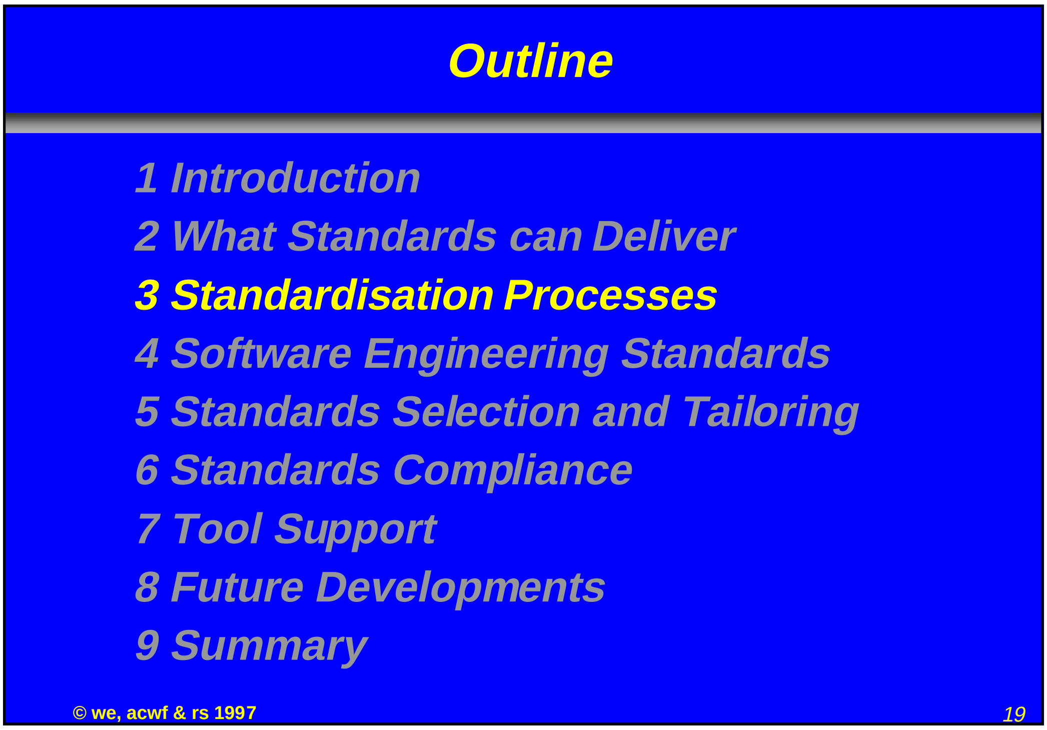
Why adopt a Standard (cont’d)

- As result of the adoption of other standards (ISO9000 and similar) or software process improvement initiatives.
- As a knock-on consequence of product certification requirements.



Development of Discipline

- Standards reflect maturation process of software engineering as a formal discipline.



Outline

1 Introduction

2 What Standards can Deliver

3 Standardisation Processes

4 Software Engineering Standards

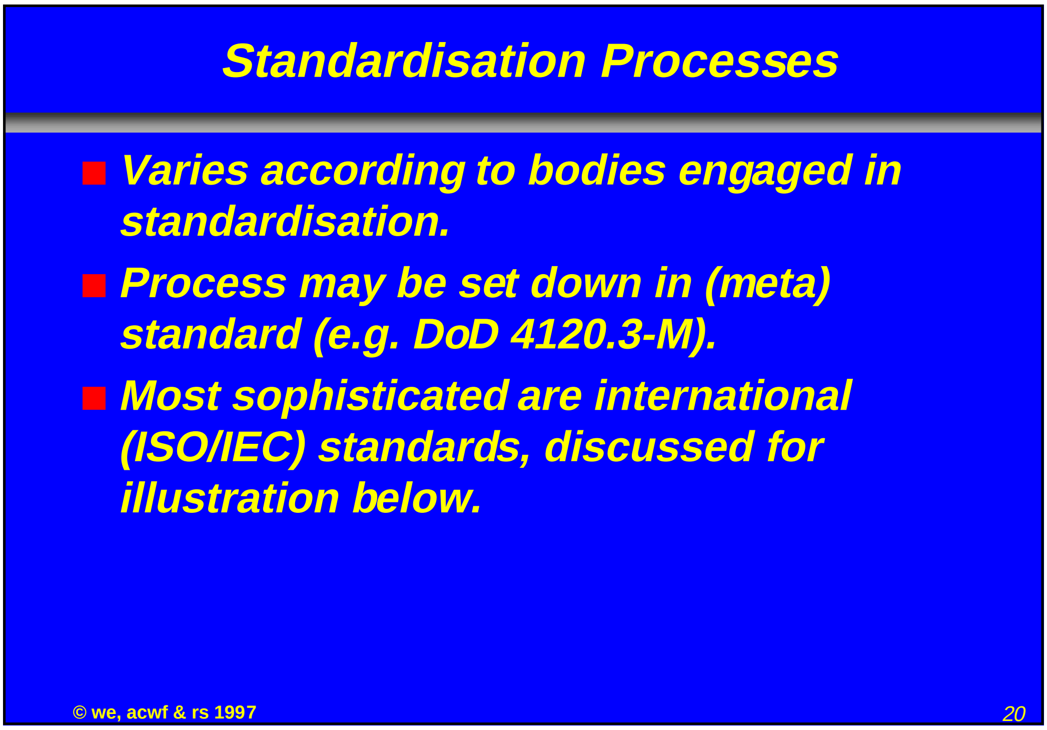
5 Standards Selection and Tailoring

6 Standards Compliance

7 Tool Support

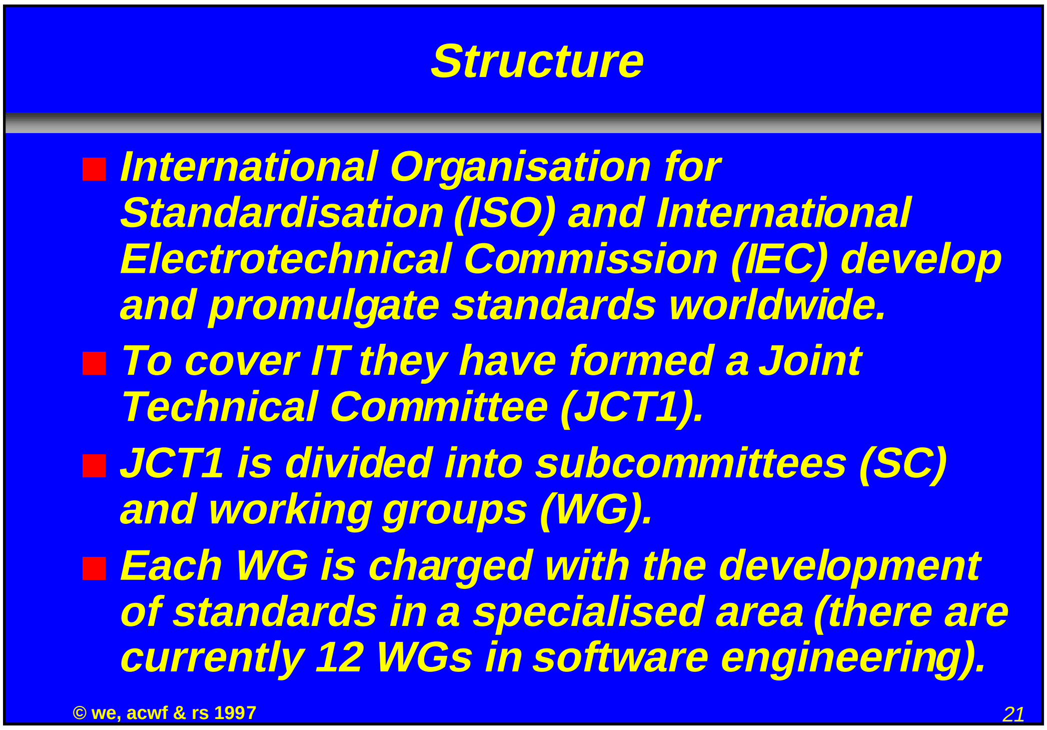
8 Future Developments

9 Summary



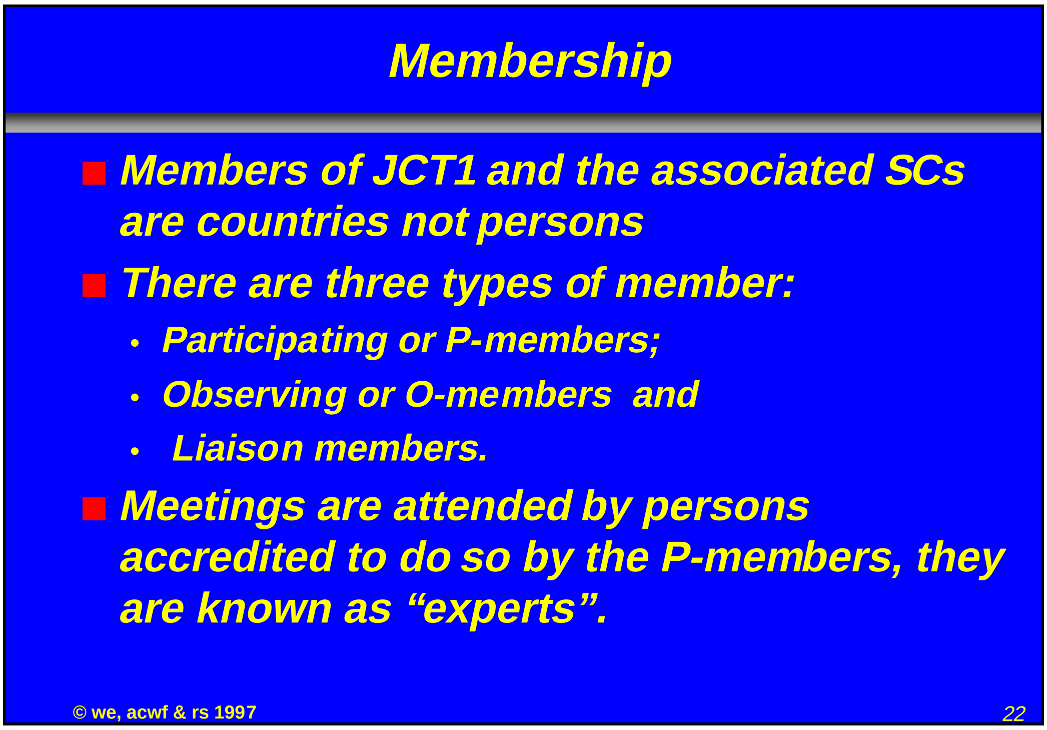
Standardisation Processes

- Varies according to bodies engaged in standardisation.
- Process may be set down in (meta) standard (e.g. DoD 4120.3-M).
- Most sophisticated are international (ISO/IEC) standards, discussed for illustration below.



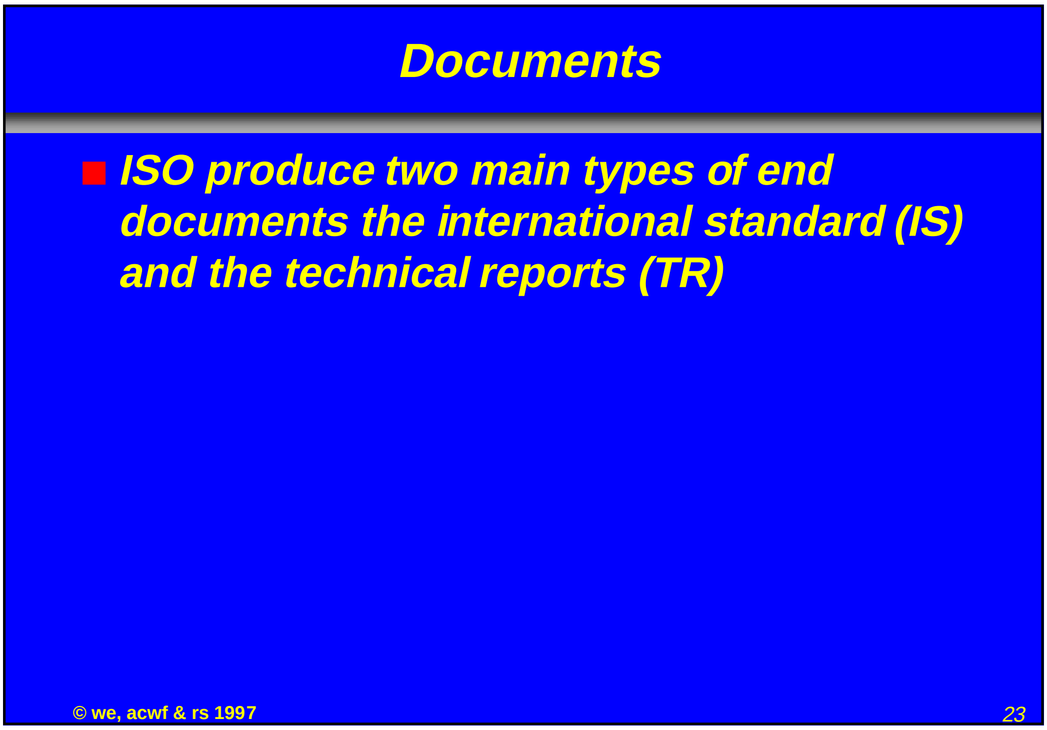
Structure

- International Organisation for Standardisation (ISO) and International Electrotechnical Commission (IEC) develop and promulgate standards worldwide.
- To cover IT they have formed a Joint Technical Committee (JCT1).
- JCT1 is divided into subcommittees (SC) and working groups (WG).
- Each WG is charged with the development of standards in a specialised area (there are currently 12 WGs in software engineering).



Membership

- Members of JCT1 and the associated SCs are countries not persons
- There are three types of member:
  - Participating or P-members;
  - Observing or O-members and
  - Liaison members.
- Meetings are attended by persons accredited to do so by the P-members, they are known as “experts”.

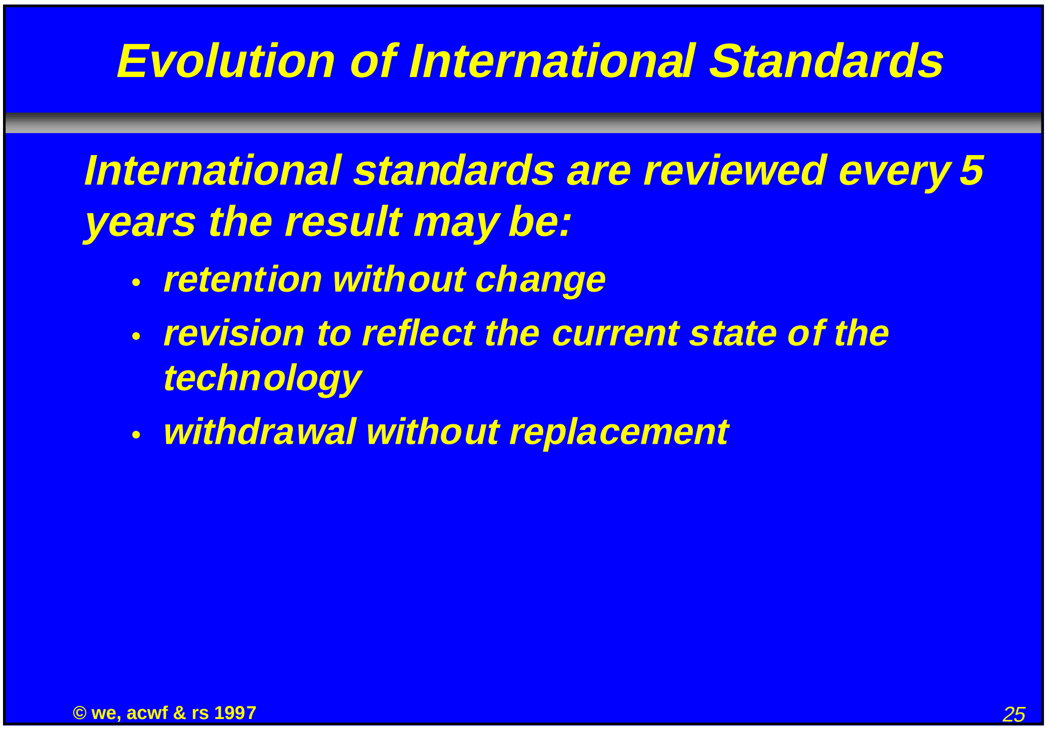


Documents

- ISO produce two main types of end documents the international standard (IS) and the technical reports (TR)

International Standard

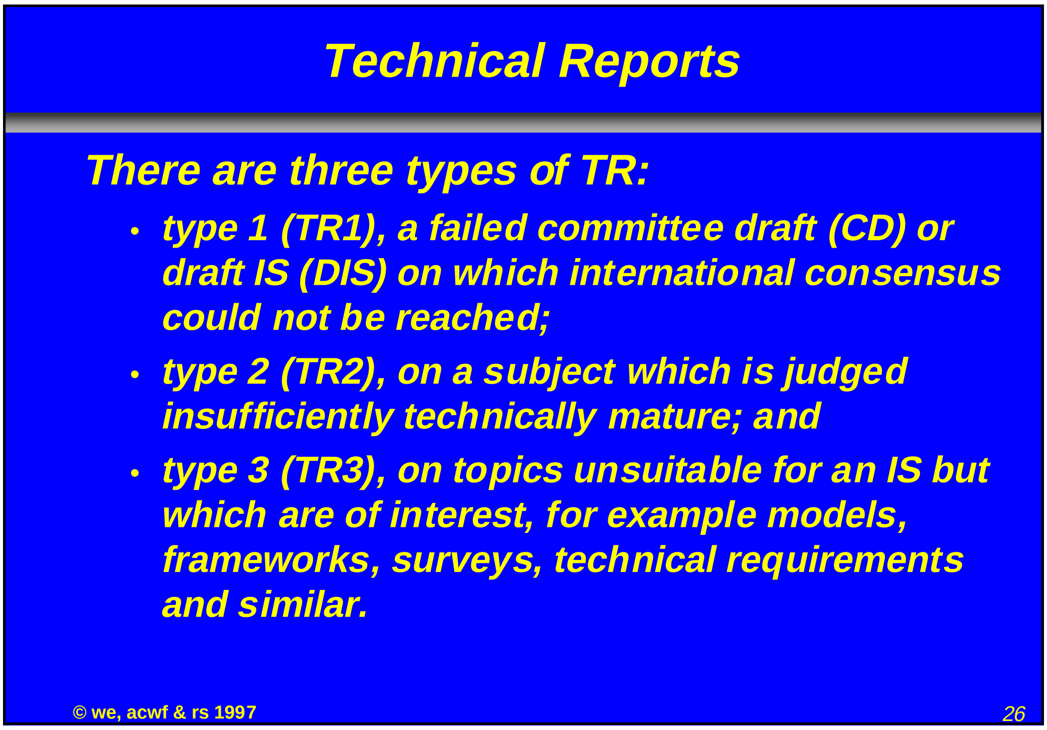
- “The social and economic long-term benefits of an IS should justify the total cost of preparing, adopting and maintaining the standard”.
- it must be demonstrated that the proposed standard is technically feasible, timely and unlikely either to be made obsolete quickly or to inhibit the benefits of technology to the users



Evolution of International Standards

International standards are reviewed every 5 years the result may be:

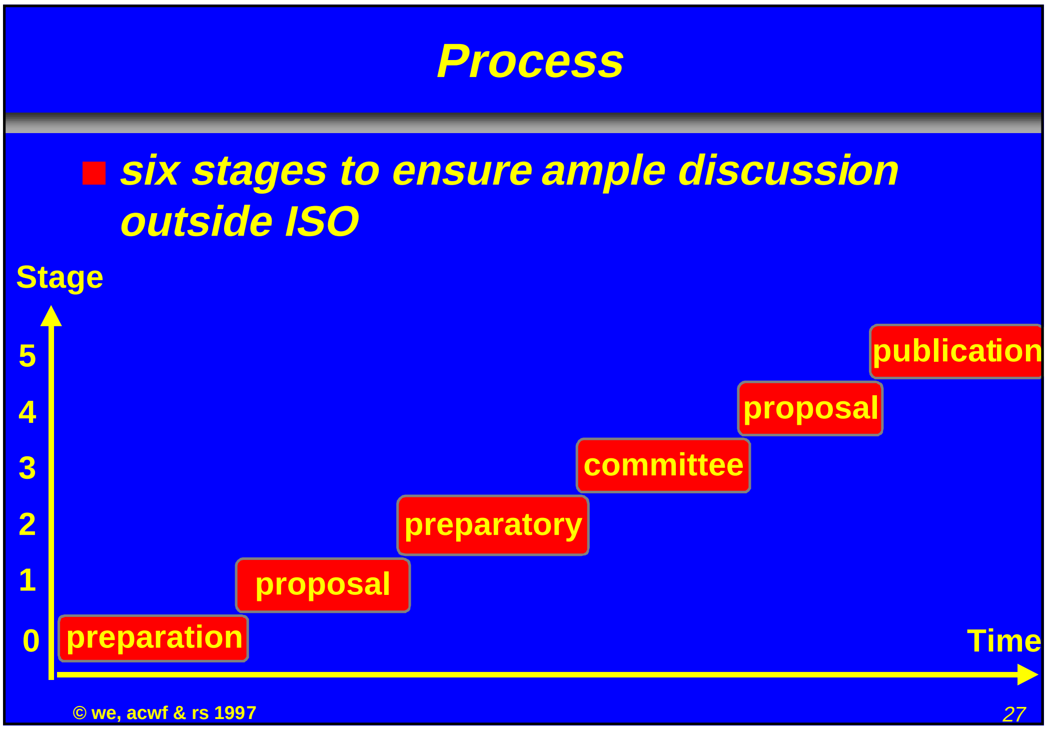
- retention without change
- revision to reflect the current state of the technology
- withdrawal without replacement



Technical Reports

There are three types of TR:

- type 1 (TR1), a failed committee draft (CD) or draft IS (DIS) on which international consensus could not be reached;
- type 2 (TR2), on a subject which is judged insufficiently technically mature; and
- type 3 (TR3), on topics unsuitable for an IS but which are of interest, for example models, frameworks, surveys, technical requirements and similar.



Process

- six stages to ensure ample discussion outside ISO

Stage

5 publication

4 proposal

3 committee

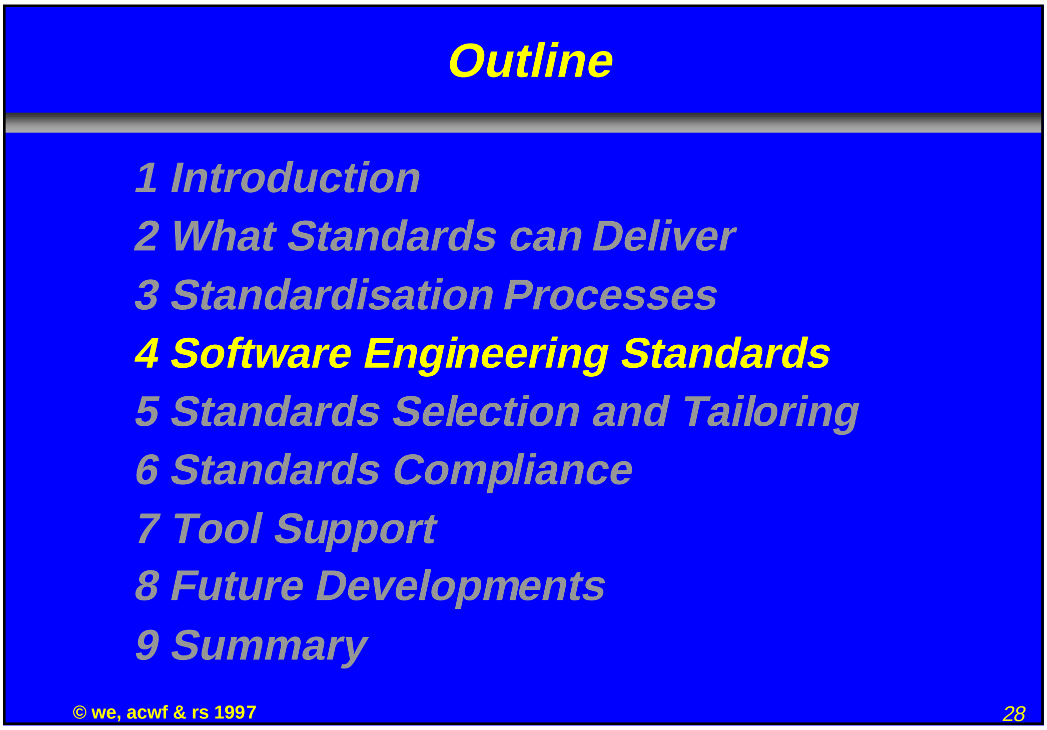
2 preparatory

1 proposal

0 preparation

Time

27



Outline

1 Introduction

2 What Standards can Deliver

3 Standardisation Processes

4 Software Engineering Standards

5 Standards Selection and Tailoring

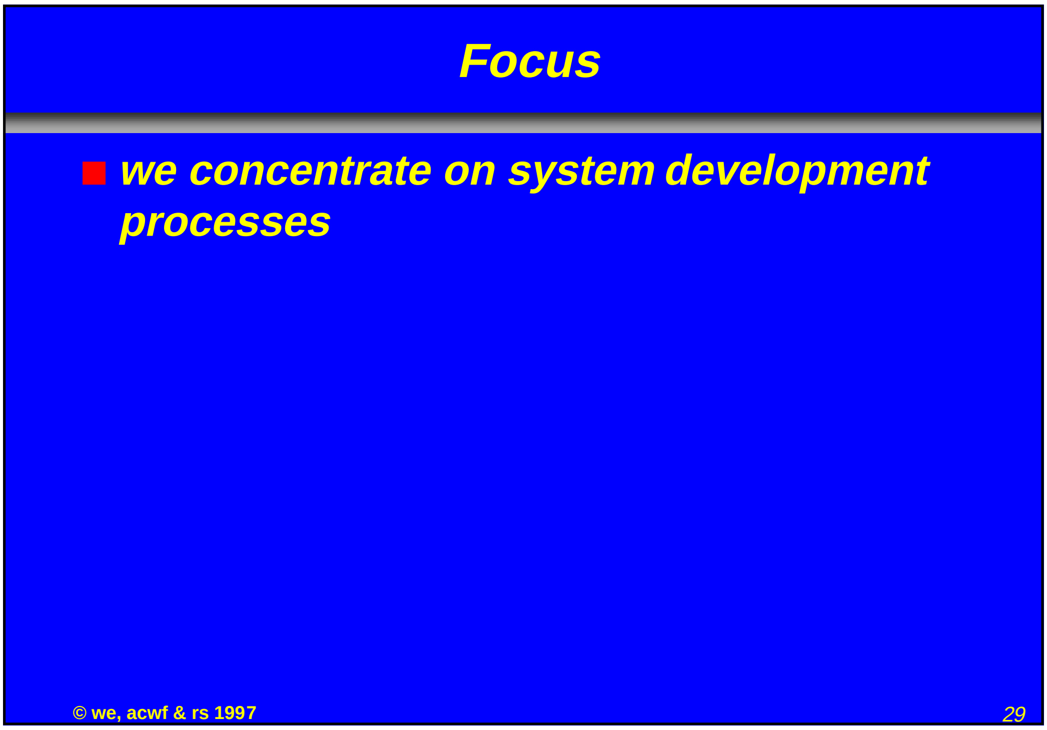
6 Standards Compliance

7 Tool Support

8 Future Developments

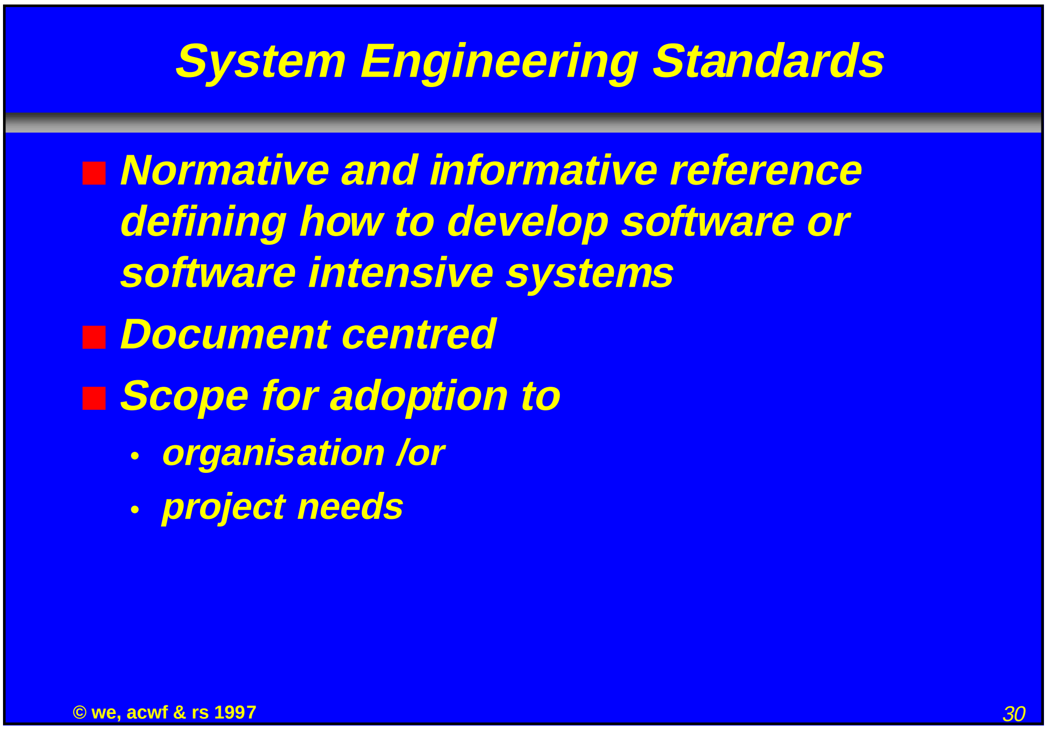
9 Summary

28



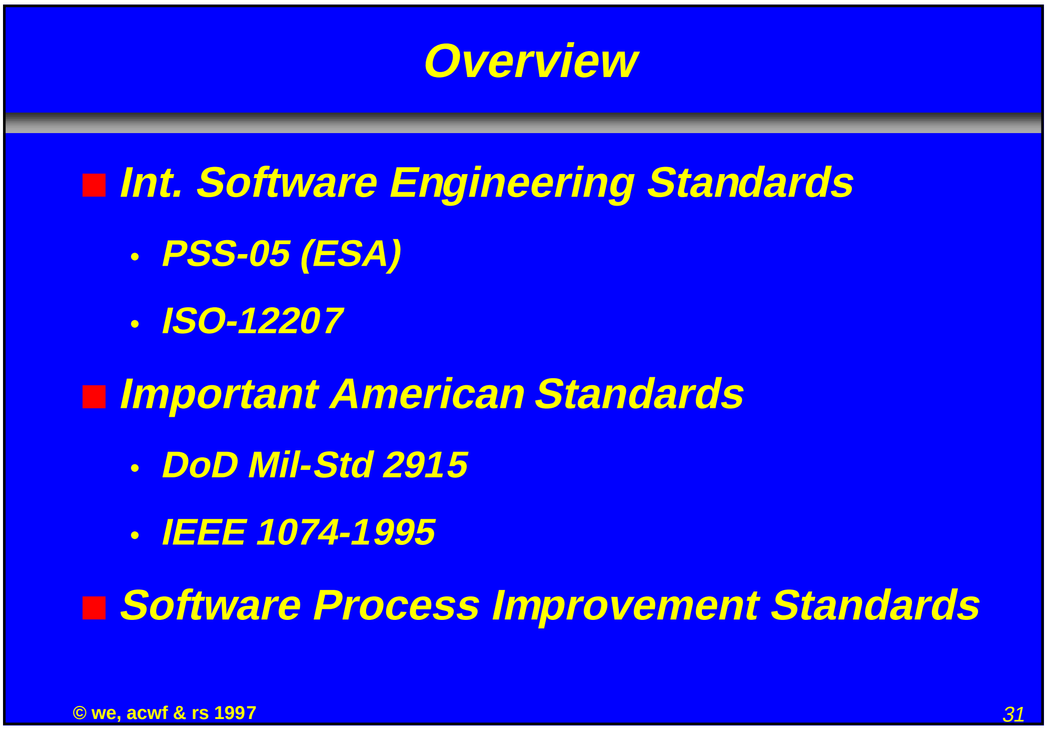
Focus

- we concentrate on system development processes



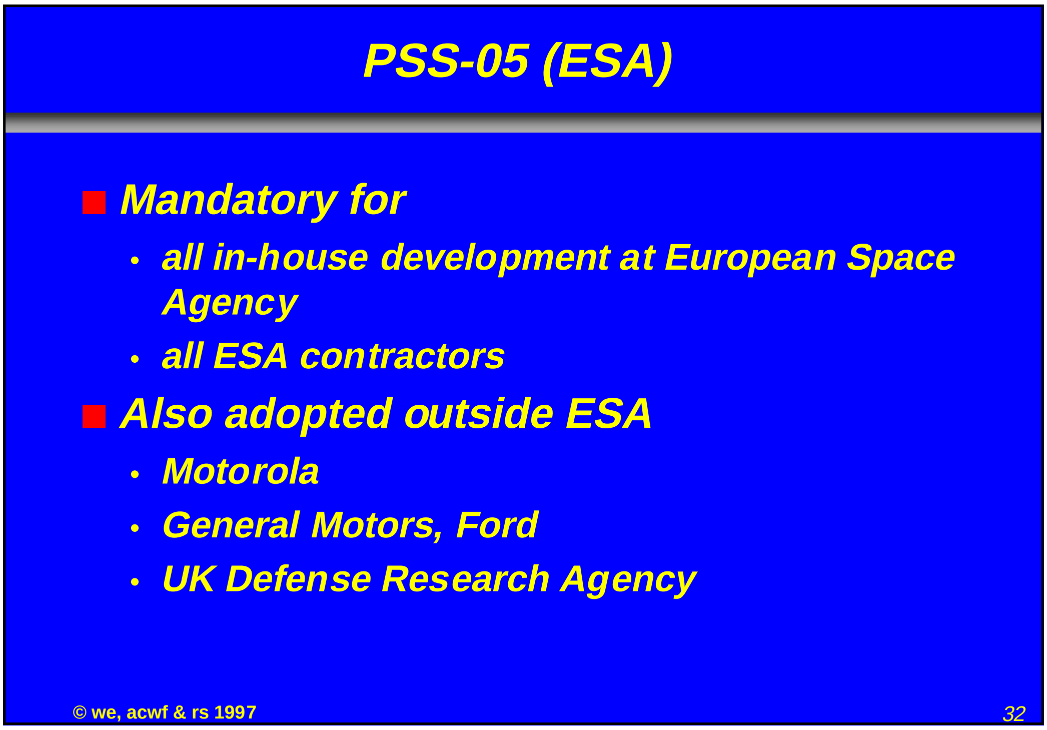
System Engineering Standards

- Normative and informative reference defining how to develop software or software intensive systems
- Document centred
- Scope for adoption to
  - organisation /or
  - project needs



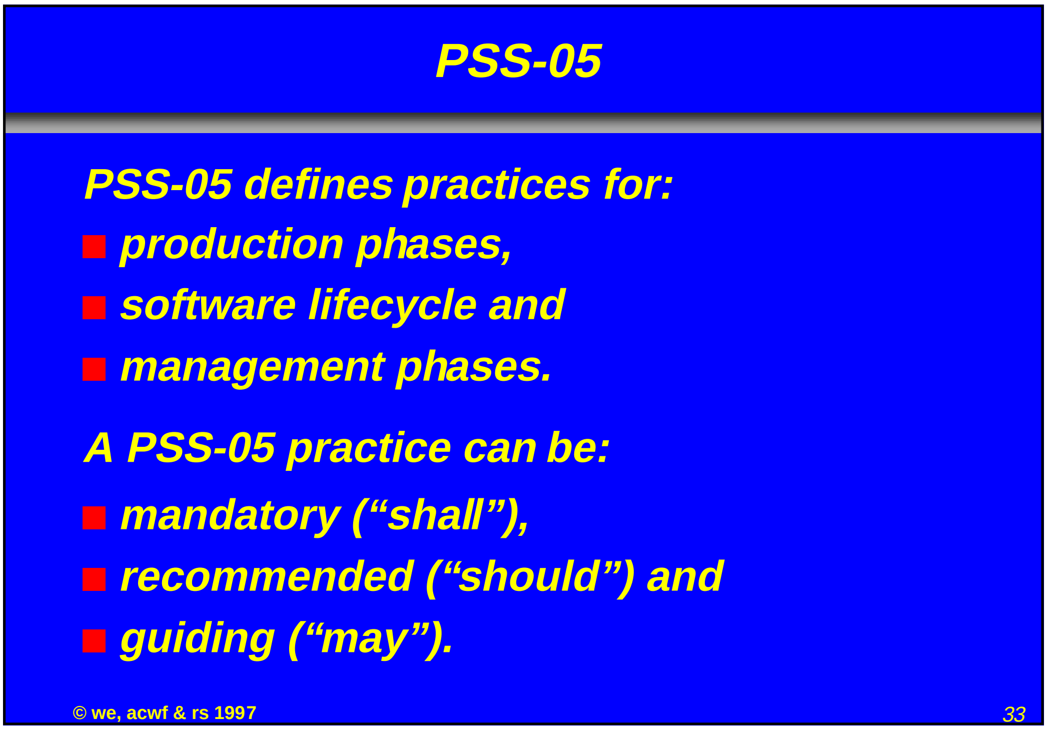
Overview

- **Int. Software Engineering Standards**
  - PSS-05 (ESA)
  - ISO-12207
- **Important American Standards**
  - DoD Mil-Std 2915
  - IEEE 1074-1995
- **Software Process Improvement Standards**



PSS-05 (ESA)

- **Mandatory for**
  - all in-house development at European Space Agency
  - all ESA contractors
- **Also adopted outside ESA**
  - Motorola
  - General Motors, Ford
  - UK Defense Research Agency



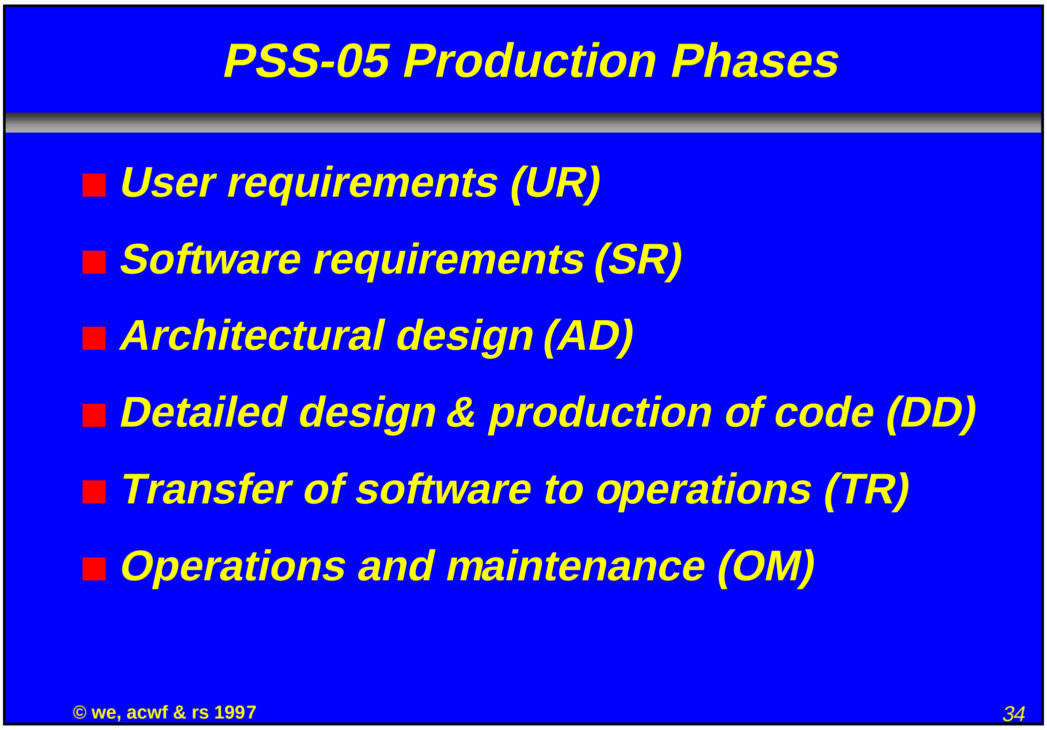
PSS-05

PSS-05 defines practices for:

- production phases,
- software lifecycle and
- management phases.

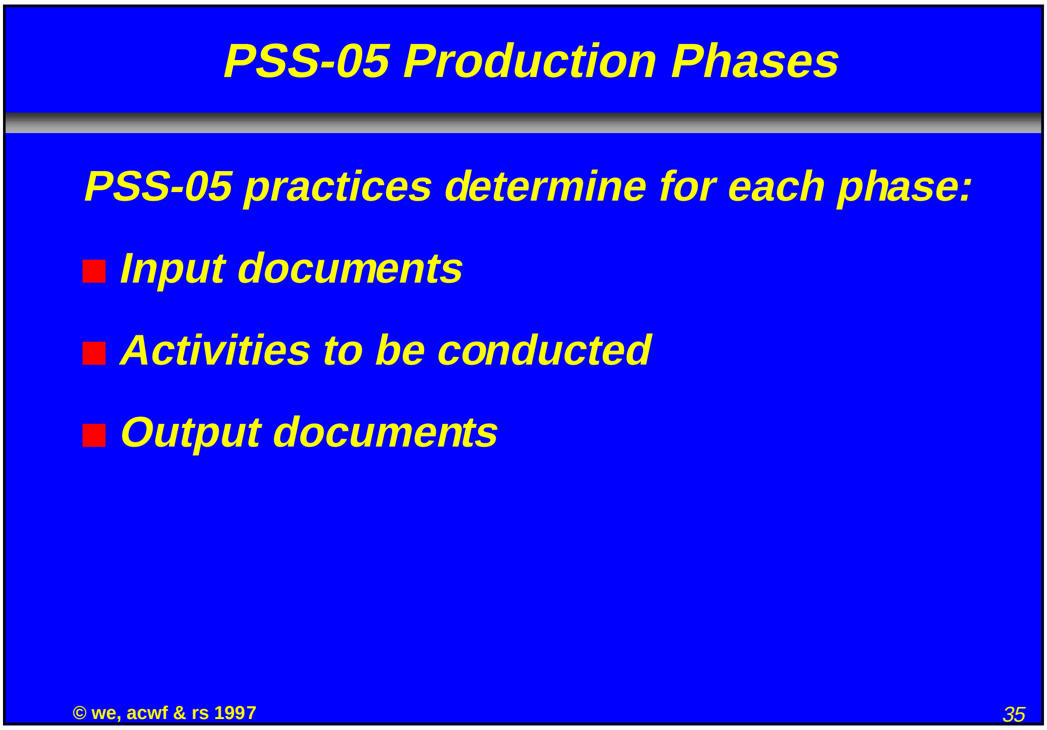
A PSS-05 practice can be:

- mandatory (“shall”),
- recommended (“should”) and
- guiding (“may”).



PSS-05 Production Phases

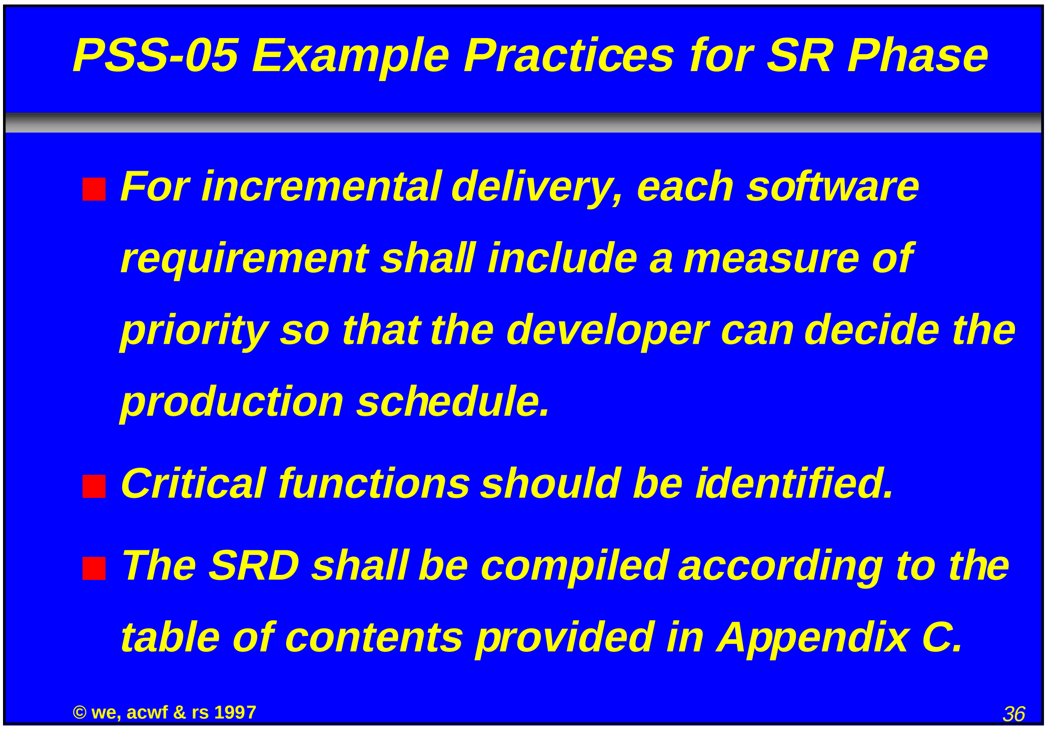
- User requirements (UR)
- Software requirements (SR)
- Architectural design (AD)
- Detailed design & production of code (DD)
- Transfer of software to operations (TR)
- Operations and maintenance (OM)



PSS-05 Production Phases

PSS-05 practices determine for each phase:

- *Input documents*
- *Activities to be conducted*
- *Output documents*



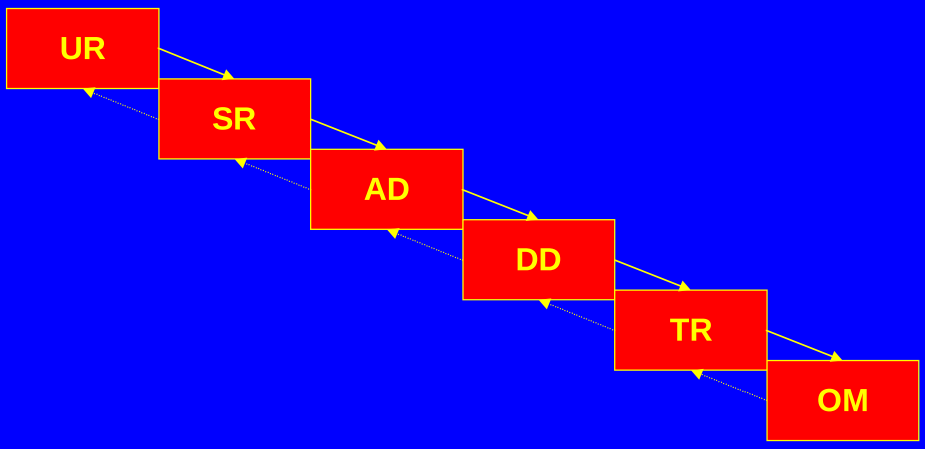
PSS-05 Example Practices for SR Phase

- *For incremental delivery, each software requirement shall include a measure of priority so that the developer can decide the production schedule.*
- *Critical functions should be identified.*
- *The SRD shall be compiled according to the table of contents provided in Appendix C.*

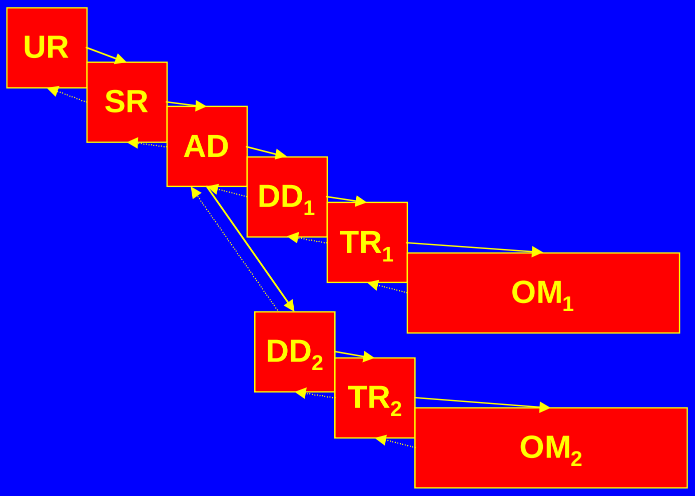
PSS-05 Software Lifecycle

- *Three lifecycle approaches are prescribed*
- *Process Engineer can select one of*
  - *Waterfall*
  - *Incremental Delivery*
  - *Evolutionary Development*

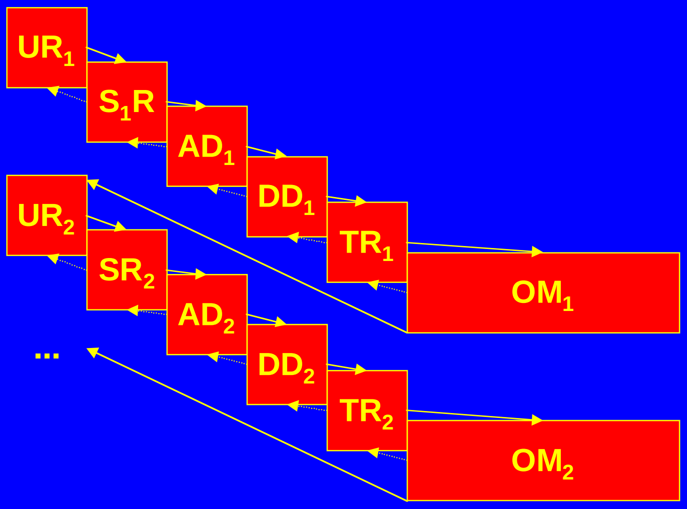
PSS-05 Waterfall Approach

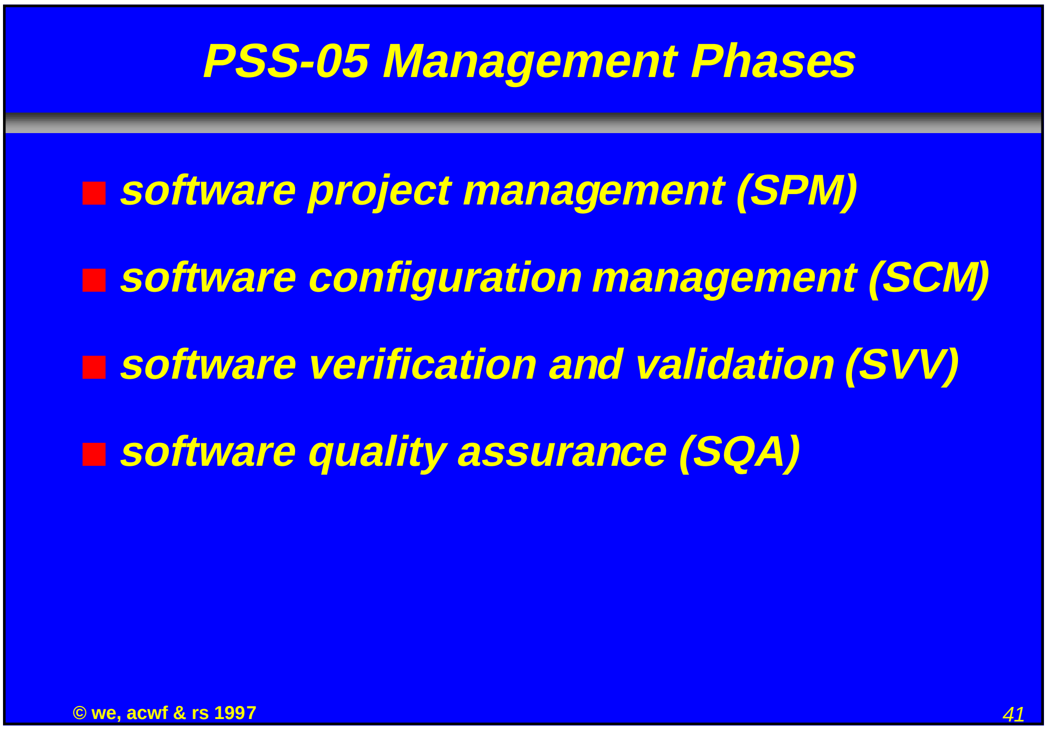


PSS-05 Incremental Delivery Approach



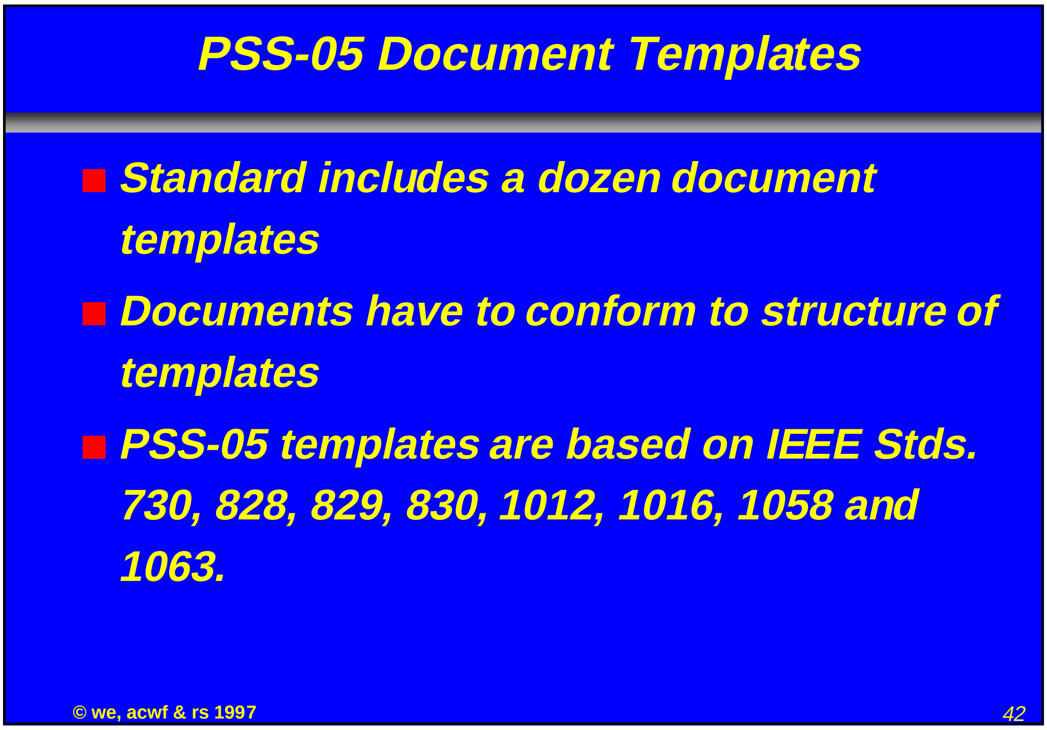
PSS-05 Evolutionary Development





PSS-05 Management Phases

- software project management (SPM)
- software configuration management (SCM)
- software verification and validation (SVV)
- software quality assurance (SQA)



PSS-05 Document Templates

- Standard includes a dozen document templates
- Documents have to conform to structure of templates
- PSS-05 templates are based on IEEE Stds. 730, 828, 829, 830, 1012, 1016, 1058 and 1063.

PSS-05 Template Sample: SRD

1 Introduction

2 General Description

2.1 Relation to current projects

Describe the relationship to other projects

…

3 Specific Requirements

List the specific requirements, with attributes. Subsections may be regrouped around high-level functions.

3.1 Functional requirements

3.2 Performance requirements

3.2 Interface requirements

3.3 Operational requirements

3.5 Resource requirements

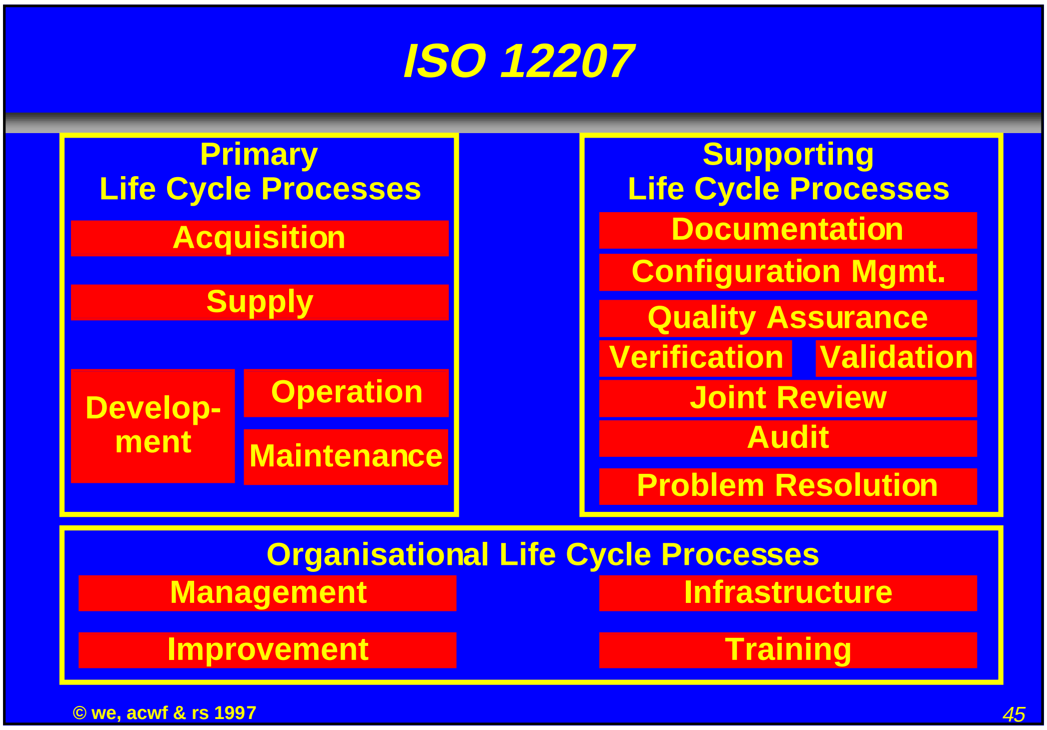
3.6 Verification requirements

…

…

ISO-12207

- ***ISO-12207 more general than PSS-05***
  - ***not only for system engineering processes***
  - ***standardises not only development activities***
  - ***allows even more adoption than PSS-05***
- ***ISO-12207 defines* tailoring *process***
- ***Areas of concern for ISO-12207***
  - ***primary life cycle activities***
  - ***supporting life cycle activities***
  - ***organisational life cycle activities***



ISO 12207

Primary Life Cycle Processes

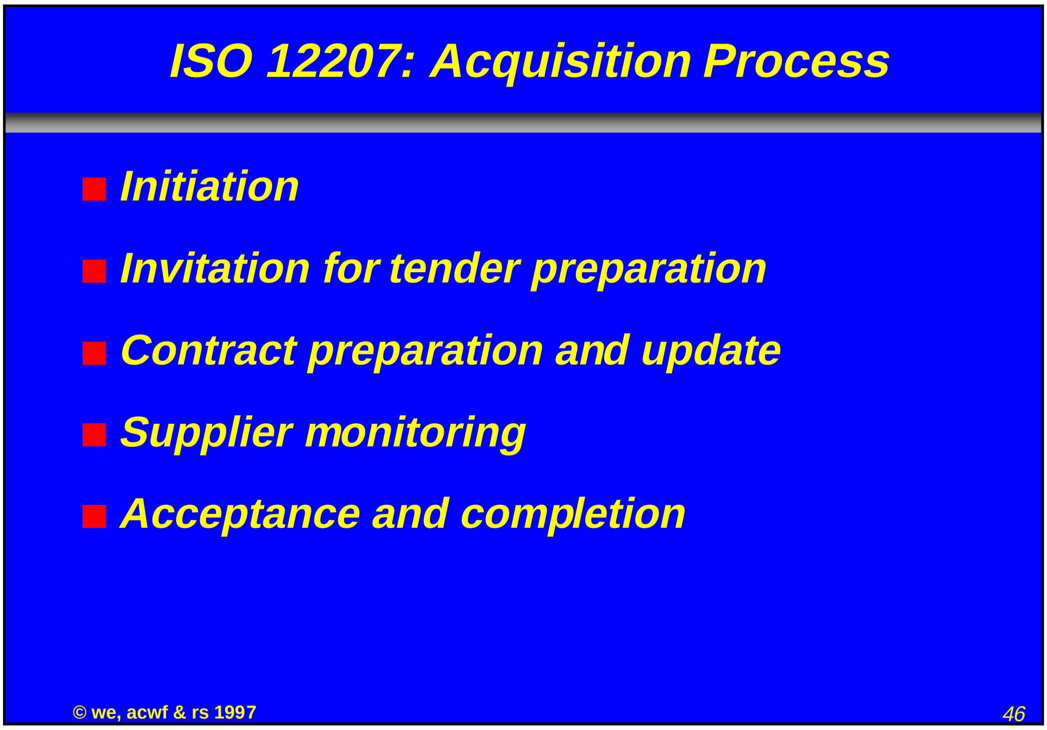
- Acquisition
- Supply
- Develop-ment
- Operation
- Maintenance

Supporting Life Cycle Processes

- Documentation
- Configuration Mgmt.
- Quality Assurance
- Verification
- Validation
- Joint Review
- Audit
- Problem Resolution

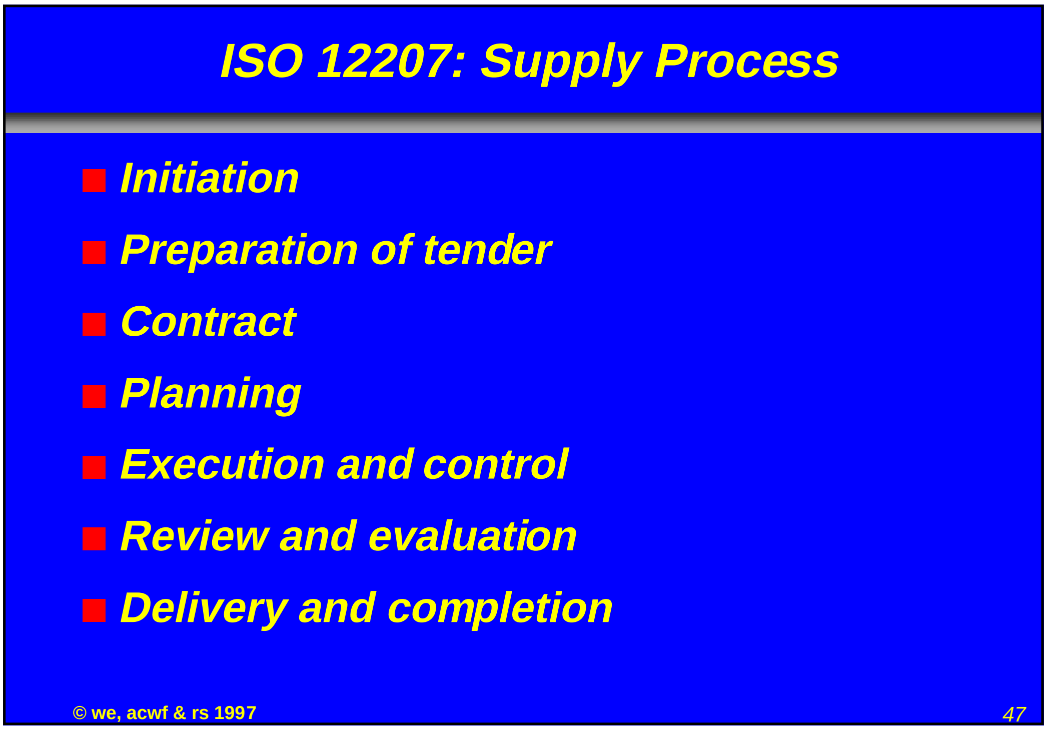
Organisational Life Cycle Processes

- Management
- Infrastructure
- Improvement
- Training



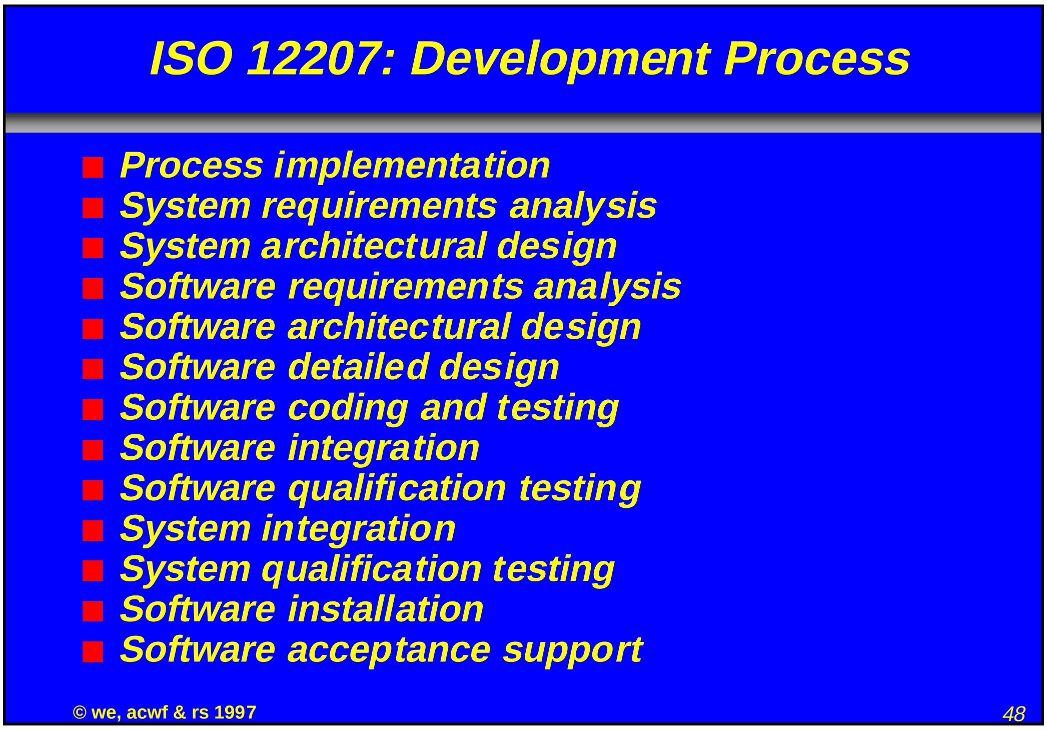
ISO 12207: Acquisition Process

- *Initiation*
- *Invitation for tender preparation*
- *Contract preparation and update*
- *Supplier monitoring*
- *Acceptance and completion*



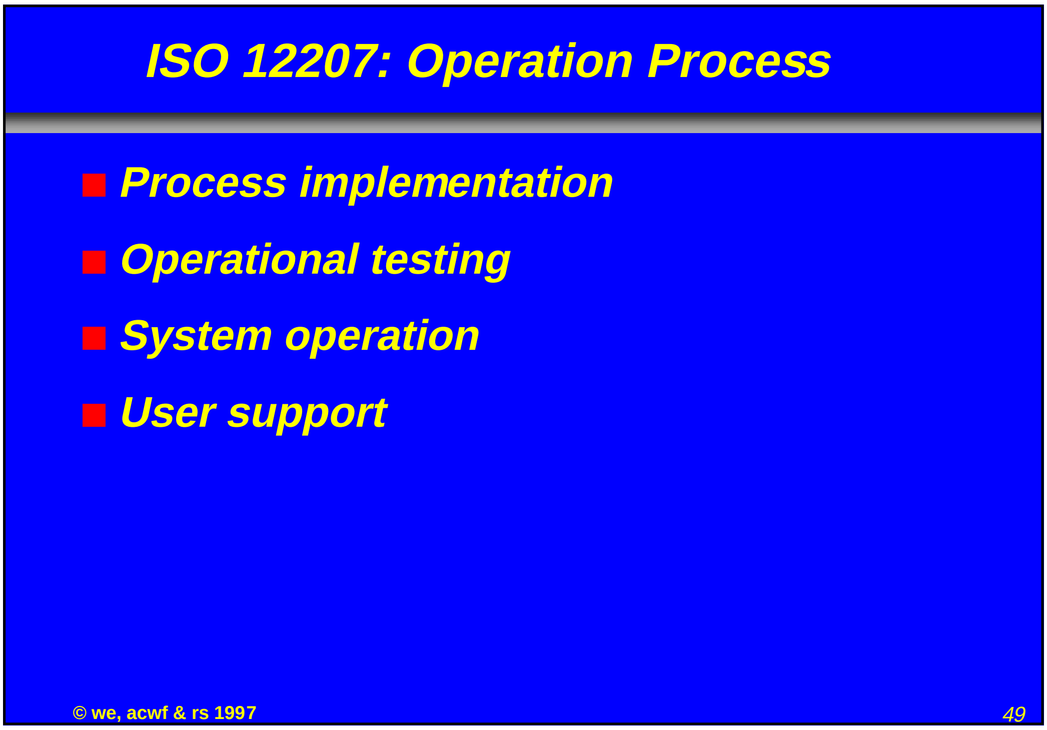
ISO 12207: Supply Process

- Initiation
- Preparation of tender
- Contract
- Planning
- Execution and control
- Review and evaluation
- Delivery and completion



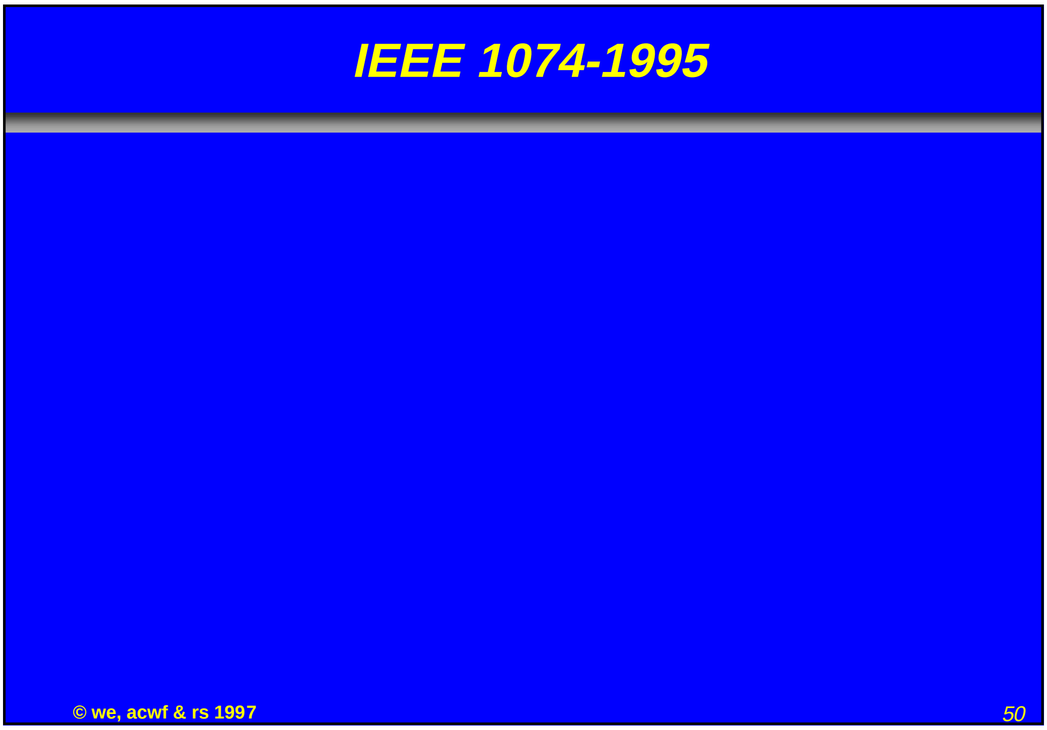
ISO 12207: Development Process

- Process implementation
- System requirements analysis
- System architectural design
- Software requirements analysis
- Software architectural design
- Software detailed design
- Software coding and testing
- Software integration
- Software qualification testing
- System integration
- System qualification testing
- Software installation
- Software acceptance support

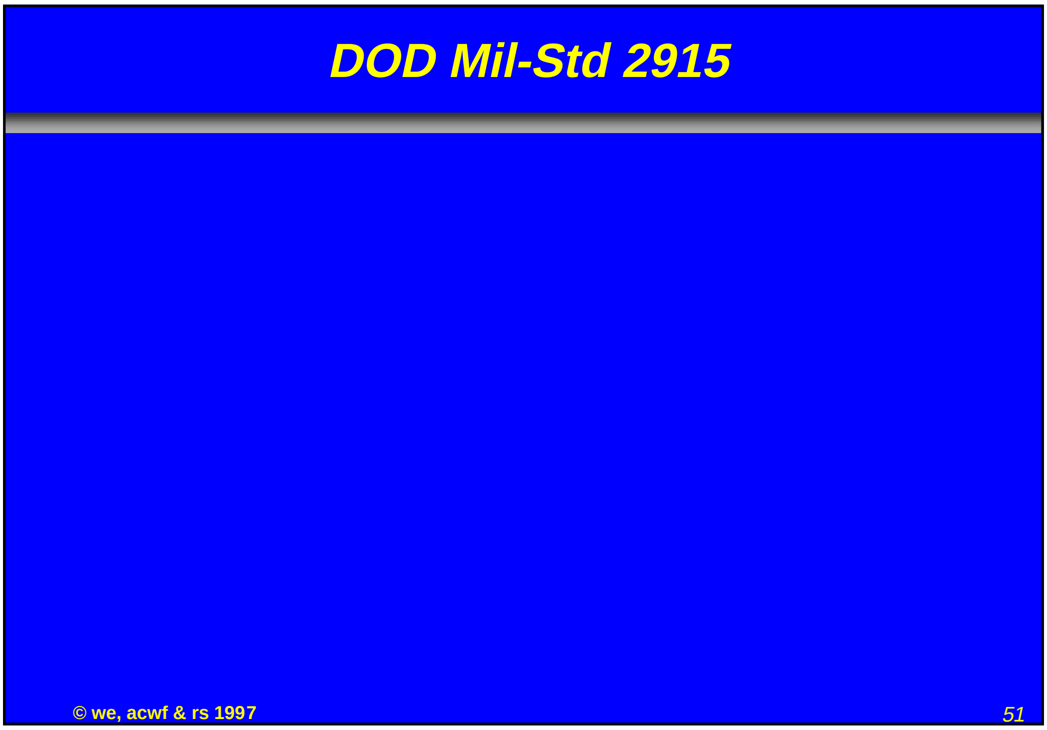


ISO 12207: Operation Process

- Process implementation
- Operational testing
- System operation
- User support



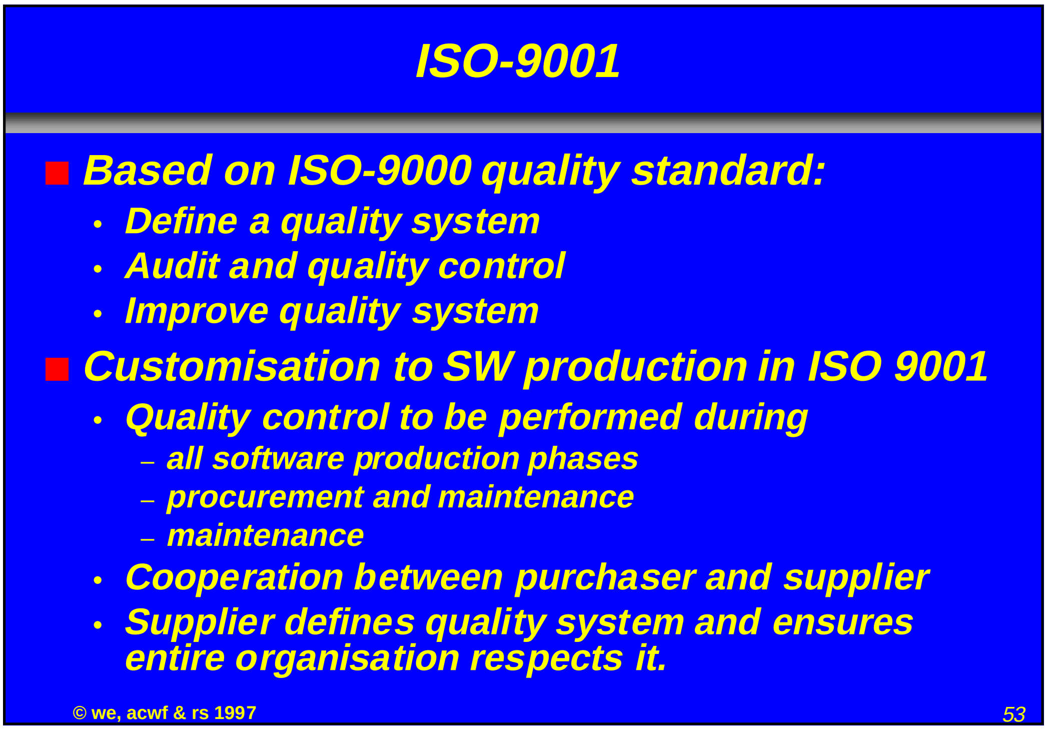
IEEE 1074-1995



DOD Mil-Std 2915

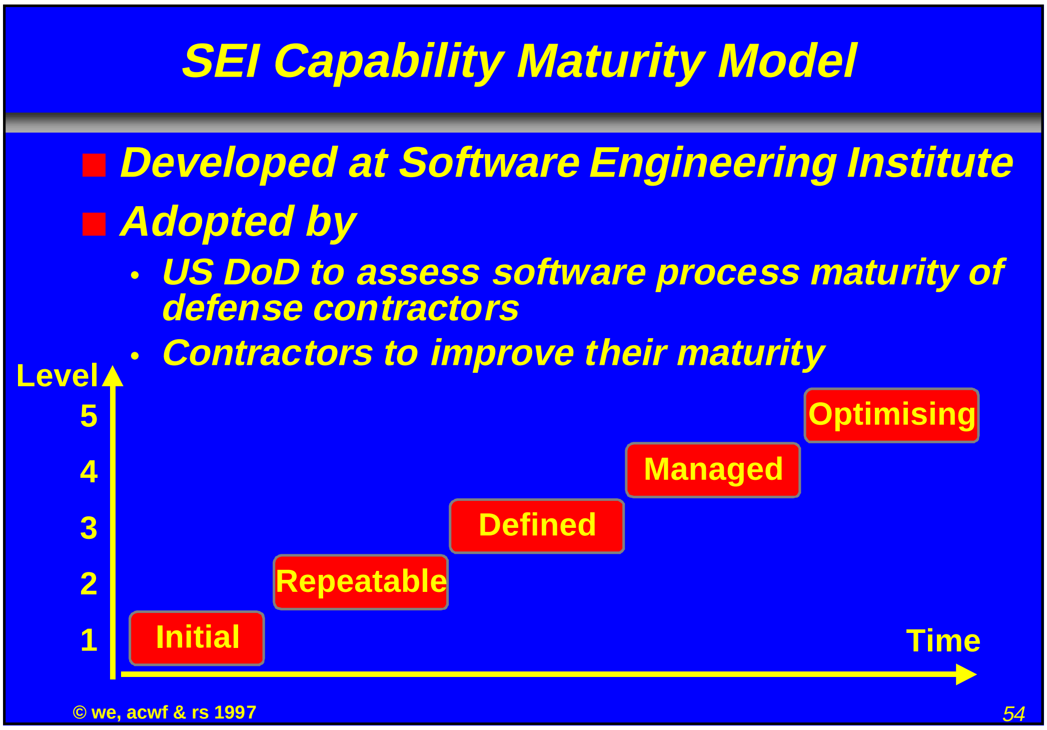
Software Process Improvement Standards

- ISO-9001
- SEI/Capability Maturity Model
- Bootstrap
- Quality Improvement Paradigm
- ISO-15504 (SPICE)



ISO-9001

- Based on ISO-9000 quality standard:
  - Define a quality system
  - Audit and quality control
  - Improve quality system
- Customisation to SW production in ISO 9001
  - Quality control to be performed during
    - all software production phases
    - procurement and maintenance
    - maintenance
  - Cooperation between purchaser and supplier
  - Supplier defines quality system and ensures entire organisation respects it.



SEI Capability Maturity Model

- Developed at Software Engineering Institute
- Adopted by
  - US DoD to assess software process maturity of defense contractors
  - Contractors to improve their maturity

Level

5 Optimising

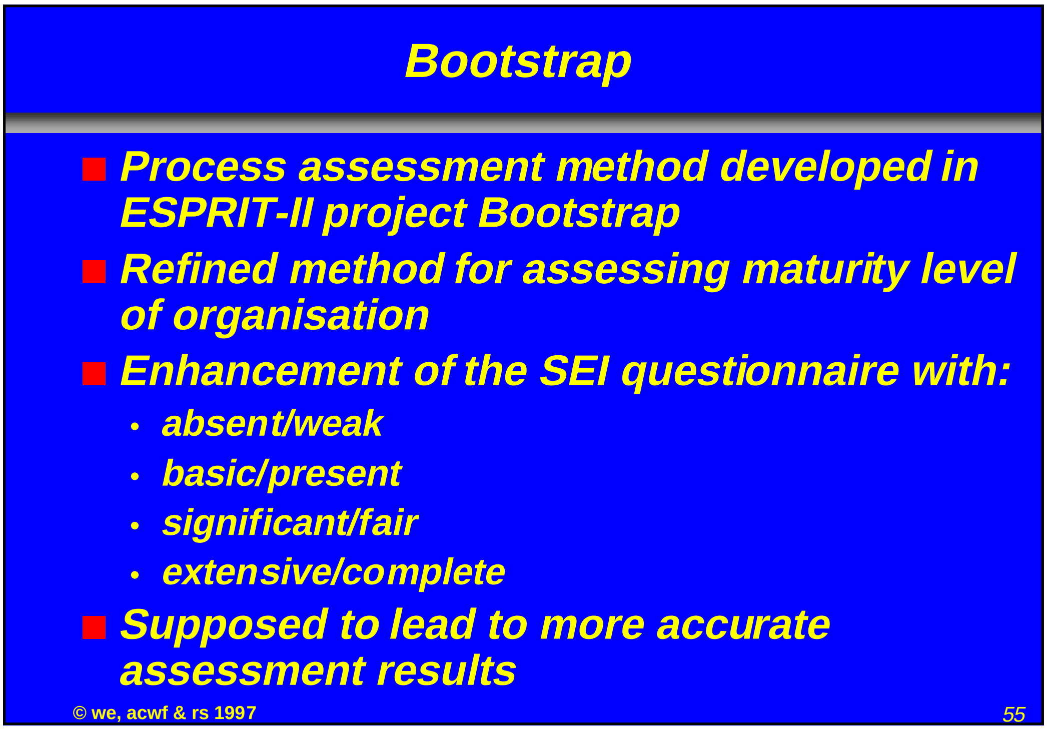
4 Managed

3 Defined

2 Repeatable

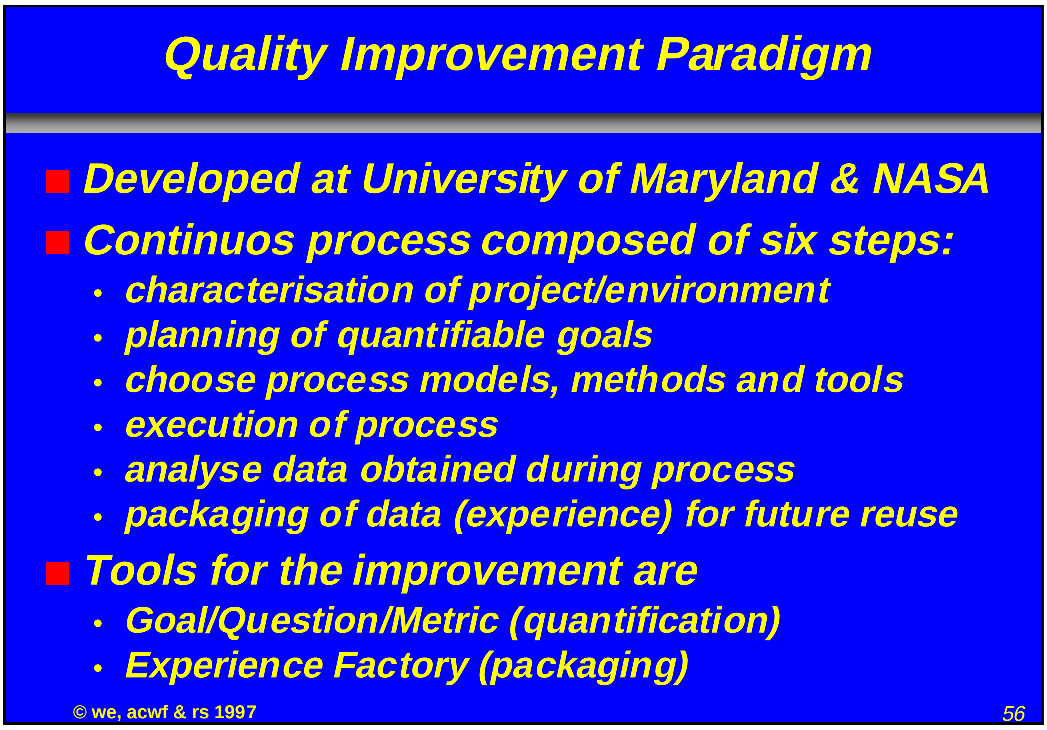
1 Initial

Time



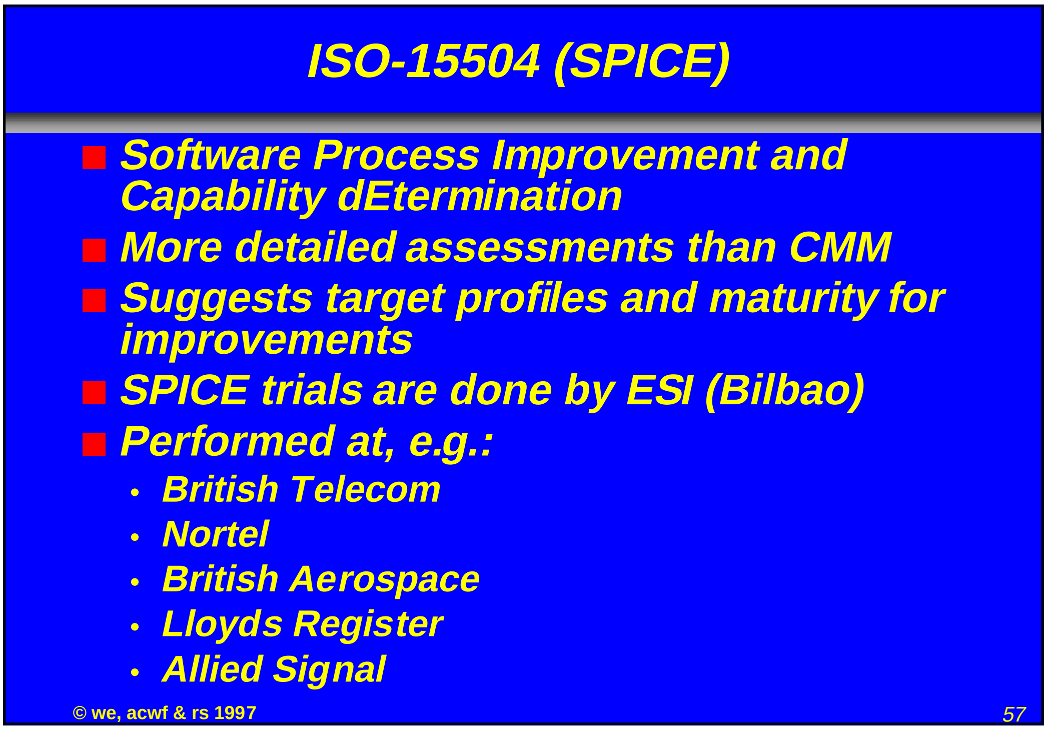
Bootstrap

- Process assessment method developed in ESPRIT-II project Bootstrap
- Refined method for assessing maturity level of organisation
- Enhancement of the SEI questionnaire with:
  - absent/weak
  - basic/present
  - significant/fair
  - extensive/complete
- Supposed to lead to more accurate assessment results



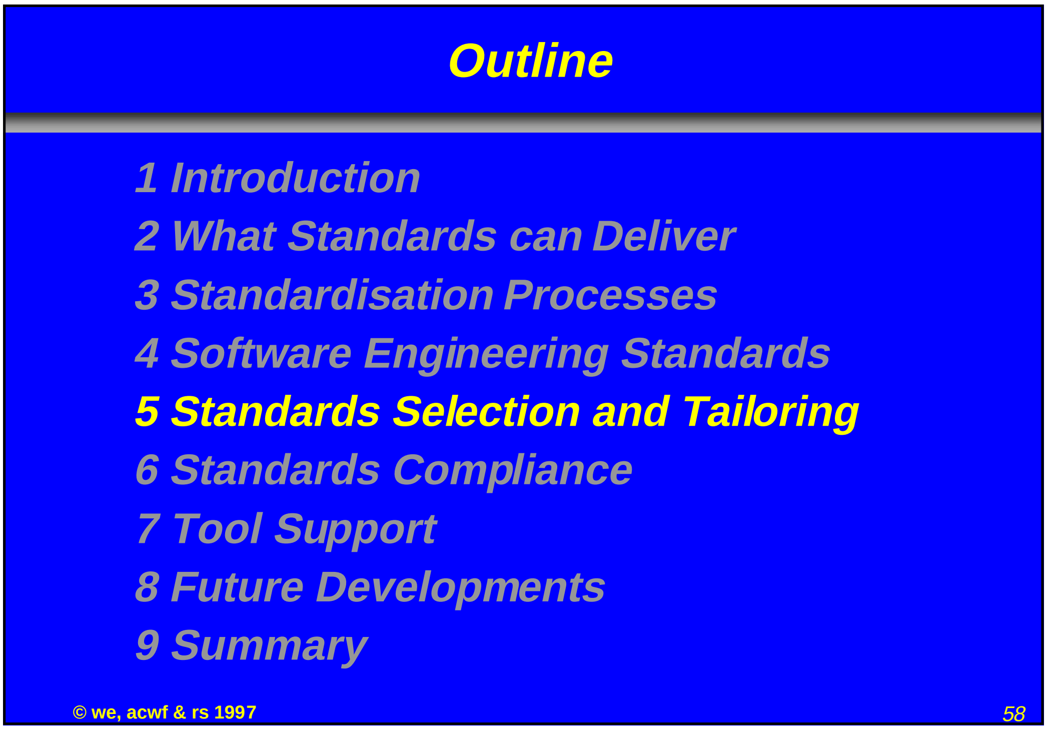
Quality Improvement Paradigm

- Developed at University of Maryland & NASA
- Continuos process composed of six steps:
  - characterisation of project/environment
  - planning of quantifiable goals
  - choose process models, methods and tools
  - execution of process
  - analyse data obtained during process
  - packaging of data (experience) for future reuse
- Tools for the improvement are
  - Goal/Question/Metric (quantification)
  - Experience Factory (packaging)



ISO-15504 (SPICE)

- Software Process Improvement and Capability dEtermination
- More detailed assessments than CMM
- Suggests target profiles and maturity for improvements
- SPICE trials are done by ESI (Bilbao)
- Performed at, e.g.:
  - British Telecom
  - Nortel
  - British Aerospace
  - Lloyds Register
  - Allied Signal



Outline

1 Introduction

2 What Standards can Deliver

3 Standardisation Processes

4 Software Engineering Standards

5 Standards Selection and Tailoring

6 Standards Compliance

7 Tool Support

8 Future Developments

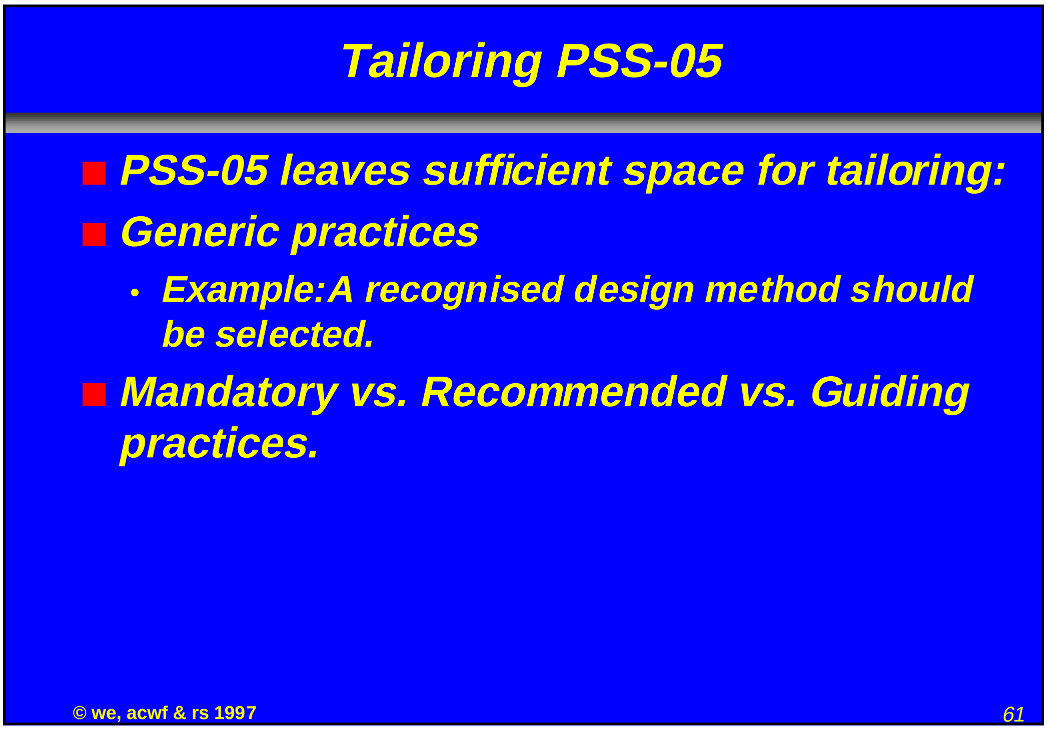
9 Summary

Selection of SW & System Eng. Standards

- *There is only a small set of internationally recognised standards.*
- *Identify key requirements for standard;*
- *Negotiate requirements with*
  - *customer*
  - *procurer*
  - *contractor;*
- *Evaluate standards against requirements;*
- *Select most appropriate standard and*
- *Tailor it*
- *Monitor use and feedback*

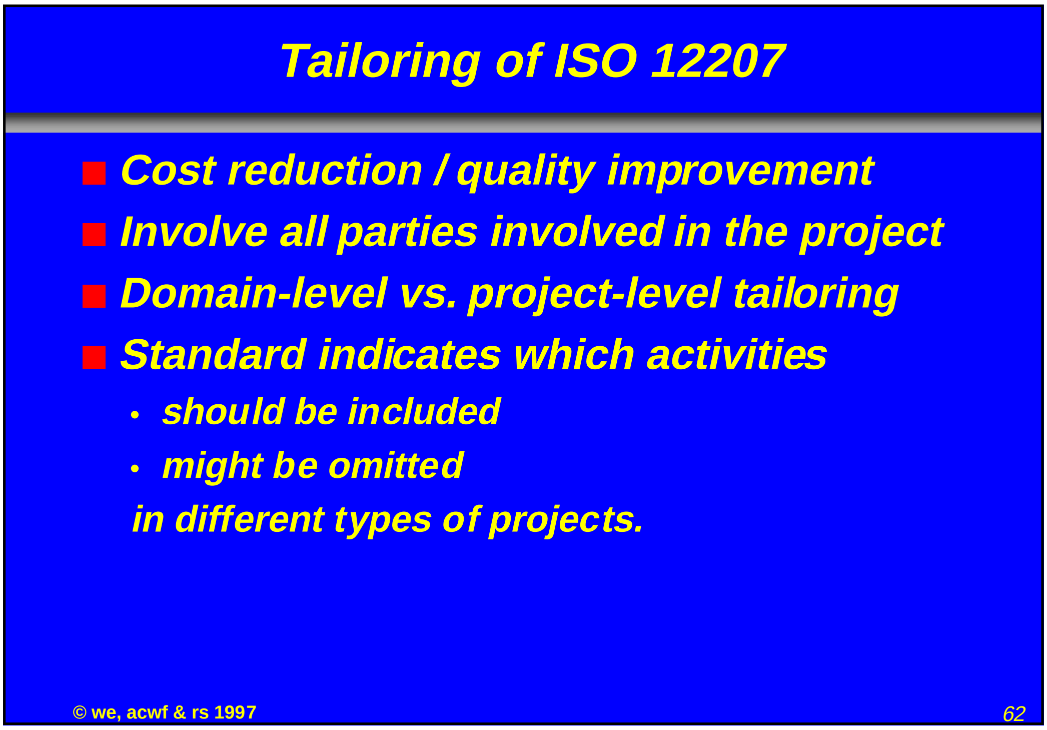
Tailoring of Standards

- *Need for Customisation*
  - *Adoption to project of different size*
  - *Integration with standards demanded by different procurers*
  - *Integration with standards used by different developers*
- *Standards leave space for tailoring*
- *Standards provide guidelines about*
  - *mandatory and optional practices*
  - *the customisation process itself*



Tailoring PSS-05

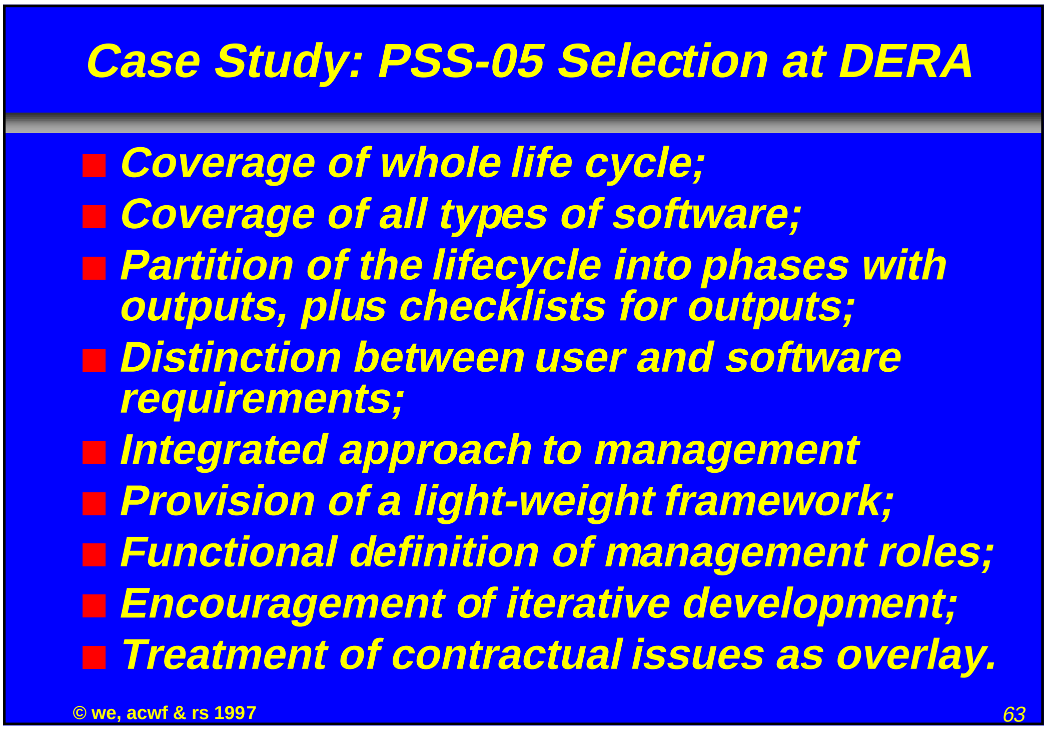
- PSS-05 leaves sufficient space for tailoring:
- Generic practices
  - Example: A recognised design method should be selected.
- Mandatory vs. Recommended vs. Guiding practices.



Tailoring of ISO 12207

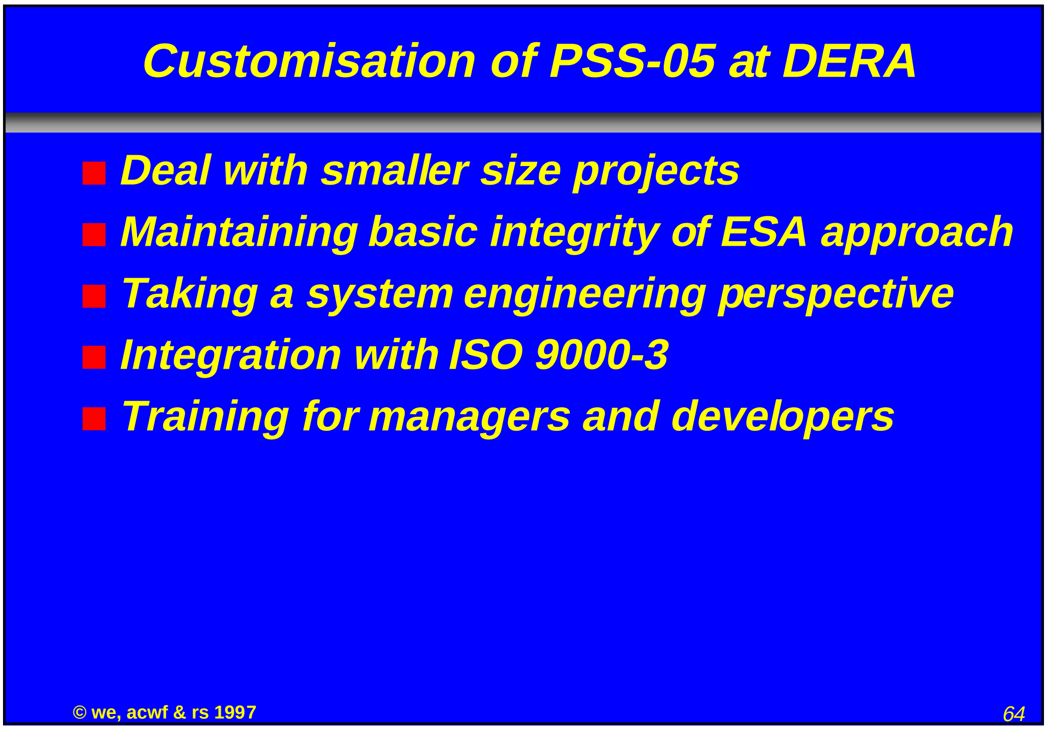
- Cost reduction / quality improvement
- Involve all parties involved in the project
- Domain-level vs. project-level tailoring
- Standard indicates which activities
  - should be included
  - might be omitted

  in different types of projects.



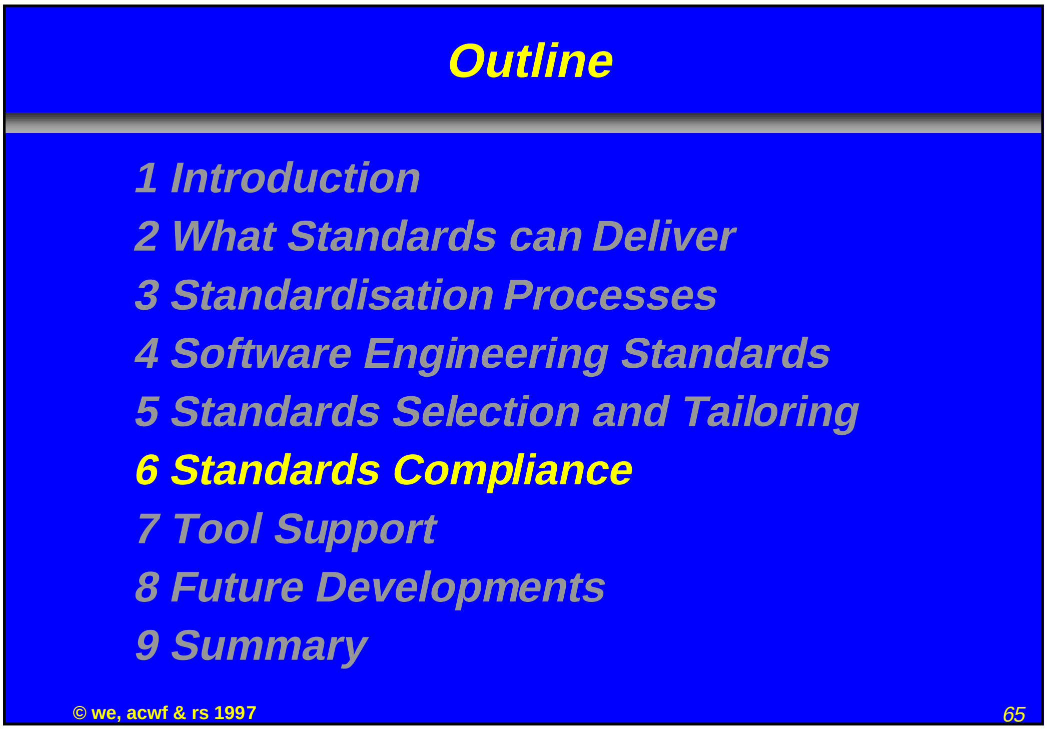
Case Study: PSS-05 Selection at DERA

- Coverage of whole life cycle;
- Coverage of all types of software;
- Partition of the lifecycle into phases with outputs, plus checklists for outputs;
- Distinction between user and software requirements;
- Integrated approach to management
- Provision of a light-weight framework;
- Functional definition of management roles;
- Encouragement of iterative development;
- Treatment of contractual issues as overlay.



Customisation of PSS-05 at DERA

- Deal with smaller size projects
- Maintaining basic integrity of ESA approach
- Taking a system engineering perspective
- Integration with ISO 9000-3
- Training for managers and developers



Outline

1 Introduction

2 What Standards can Deliver

3 Standardisation Processes

4 Software Engineering Standards

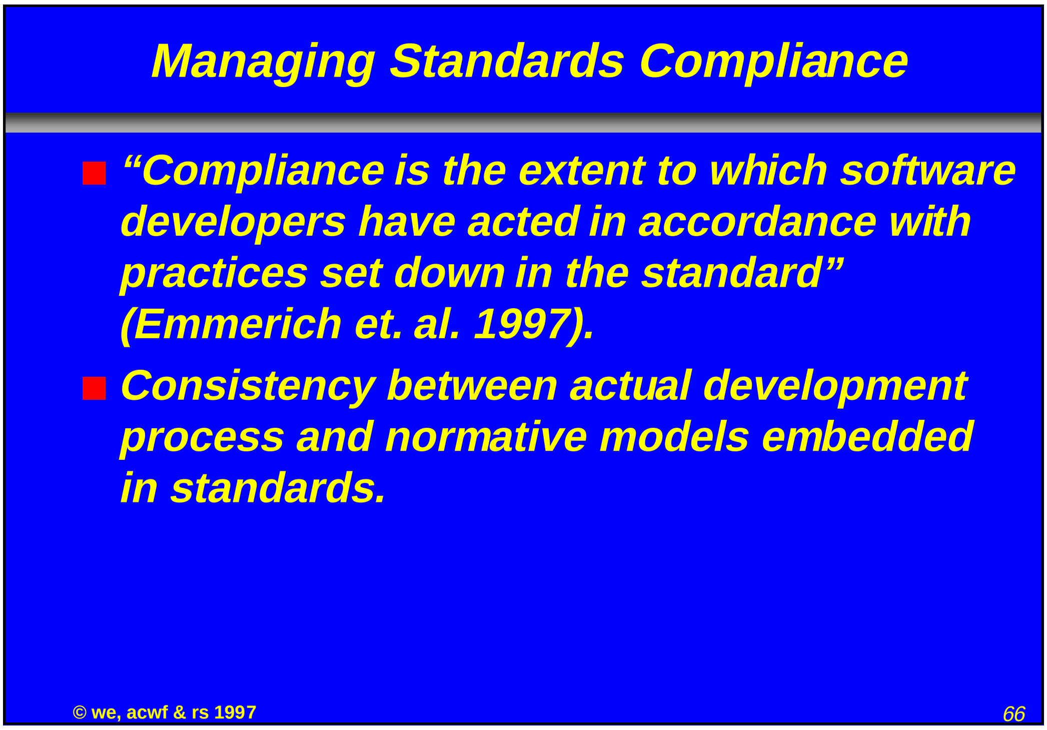
5 Standards Selection and Tailoring

6 Standards Compliance

7 Tool Support

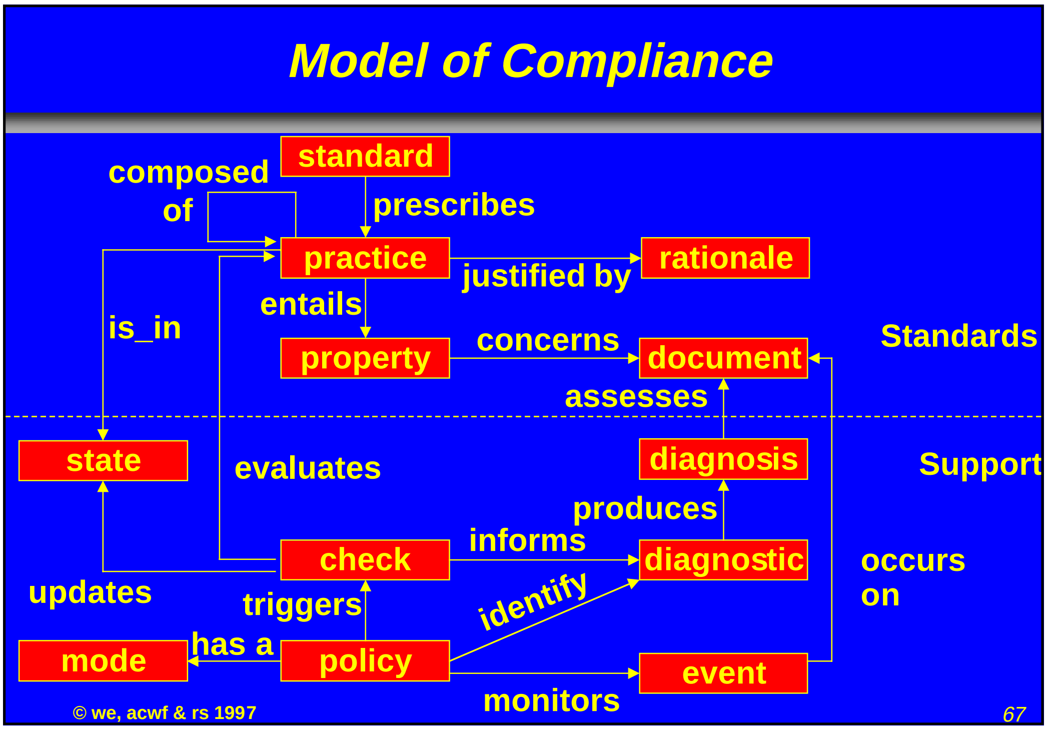
8 Future Developments

9 Summary



Managing Standards Compliance

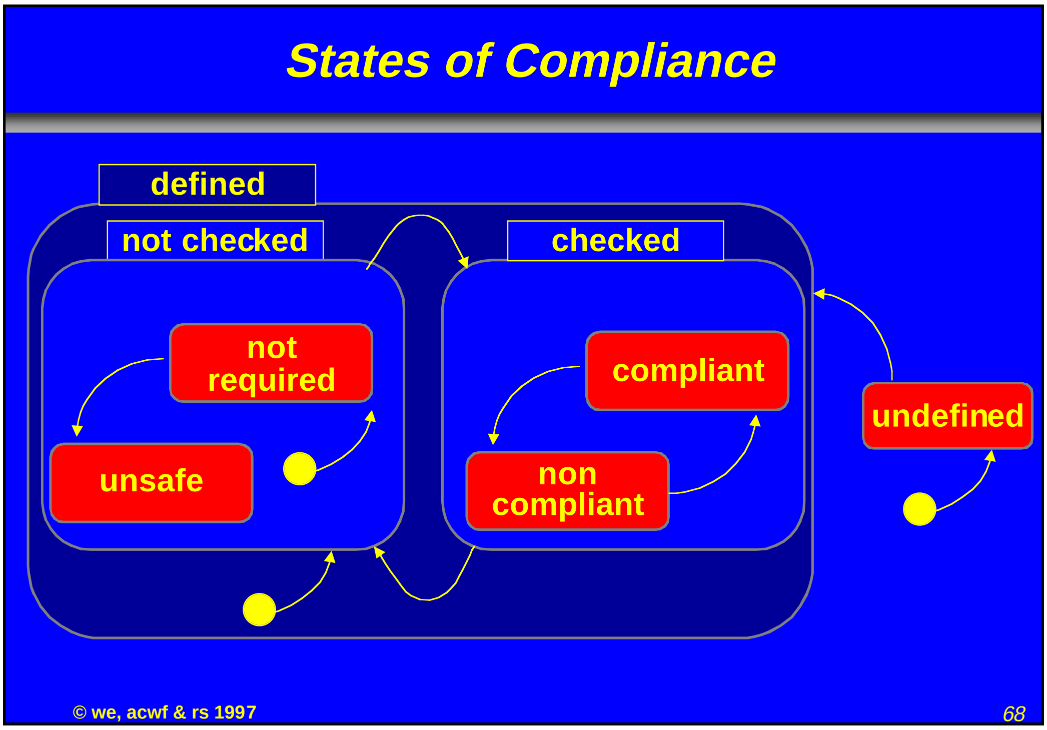
- *“Compliance is the extent to which software developers have acted in accordance with practices set down in the standard” (Emmerich et. al. 1997).*
- *Consistency between actual development process and normative models embedded in standards.*



Model of Compliance

© we, acwf & rs 1997

67



States of Compliance

© we, acwf & rs 1997

68

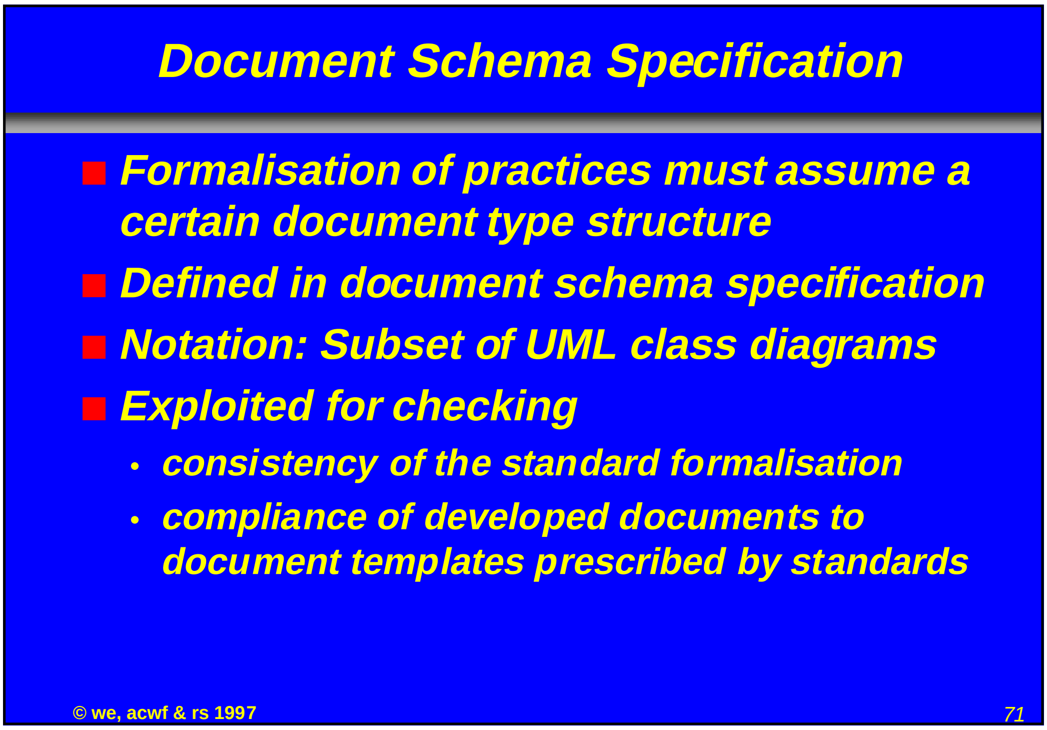
Sample Practices (PSS-05)

UR04: For incremental delivery each user requirement shall include a measure of priority so that the developer can decide the production schedule.

UR10: An output of the User Requirements phase shall be the User Requirements Document.

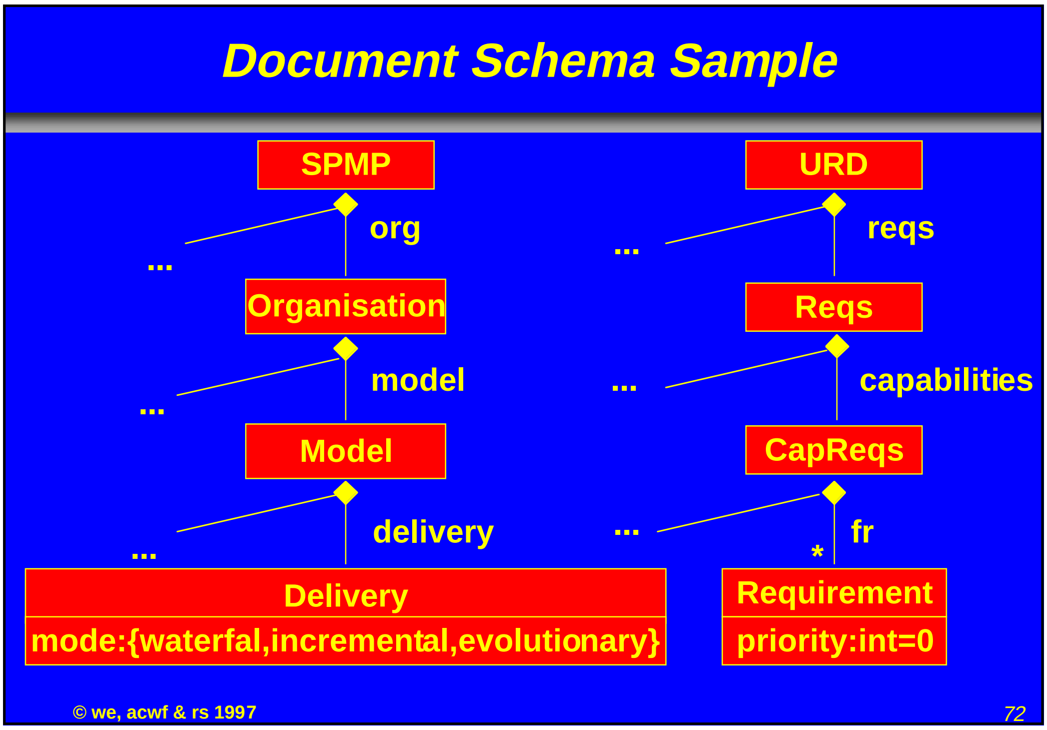
Policy Modes

- Error: Prevent the developer from completing the action that would result in non-compliance
- Warning: Indicate to the developer that the result of the action is non-compliance
- Guideline: Suggest to the developer that compliance to a practice should be checked



Document Schema Specification

- Formalisation of practices must assume a certain document type structure
- Defined in document schema specification
- Notation: Subset of UML class diagrams
- Exploited for checking
  - consistency of the standard formalisation
  - compliance of developed documents to document templates prescribed by standards



Document Schema Sample

SPMP

org

...

Organisation

model

...

Model

delivery

...

Delivery

mode:{waterfal,incremental,evolutionary}

URD

reqs

...

Reqs

capabilities

...

CapReqs

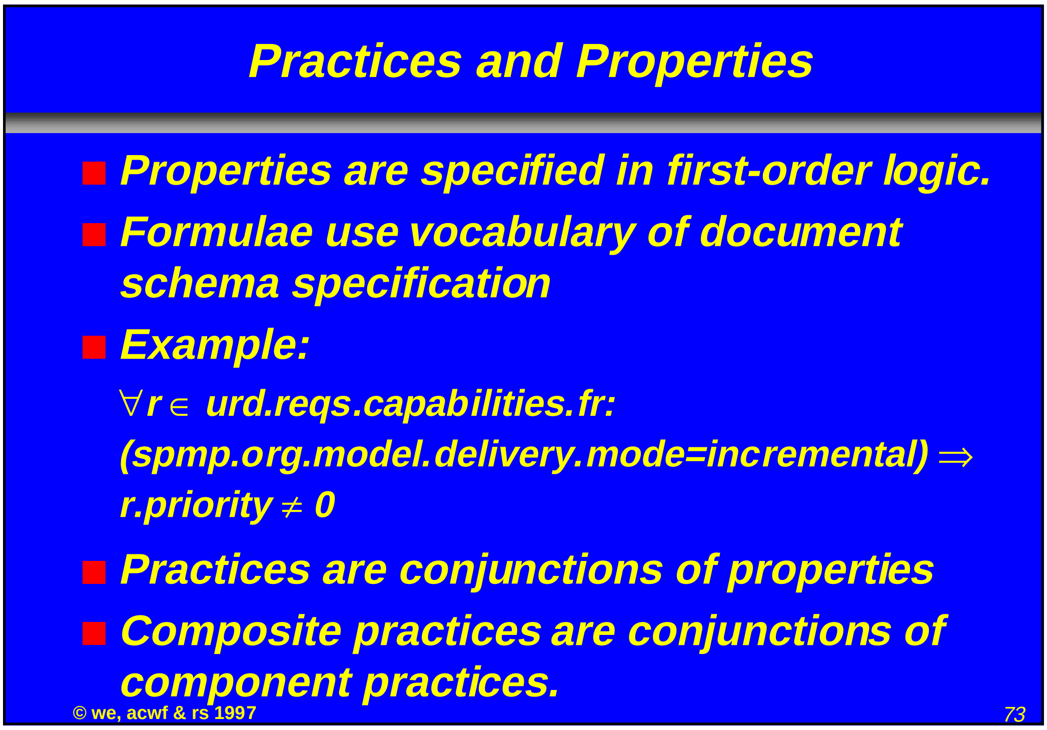
fr

...

*

Requirement

priority:int=0

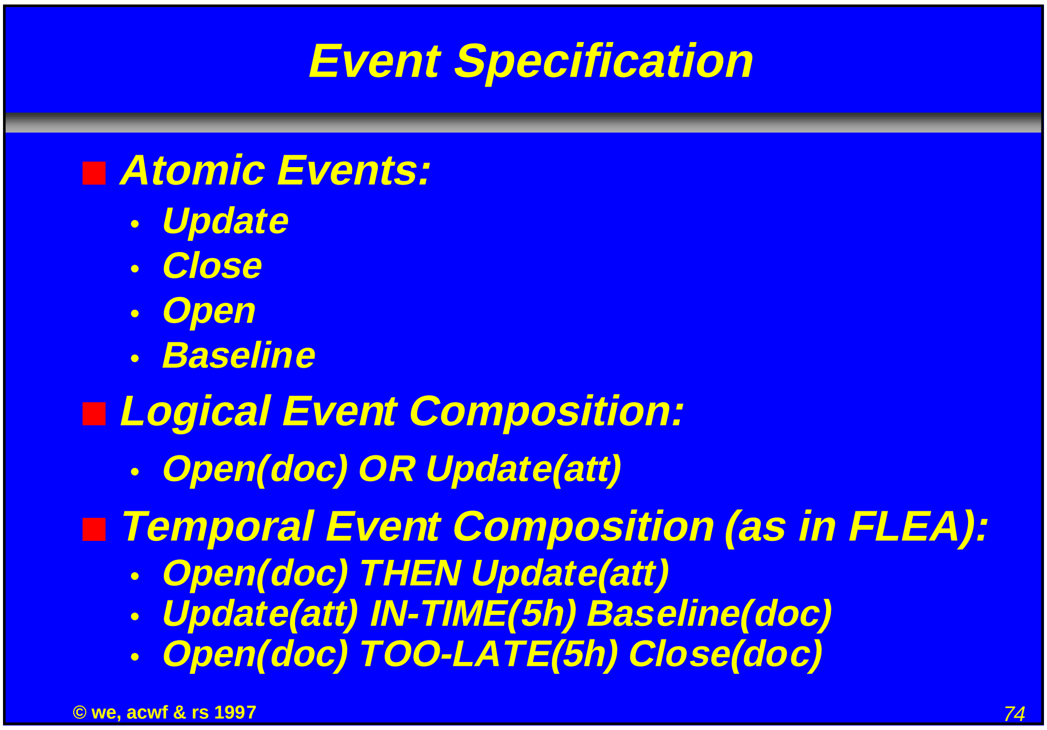


Practices and Properties

- Properties are specified in first-order logic.
- Formulae use vocabulary of document schema specification
- Example:

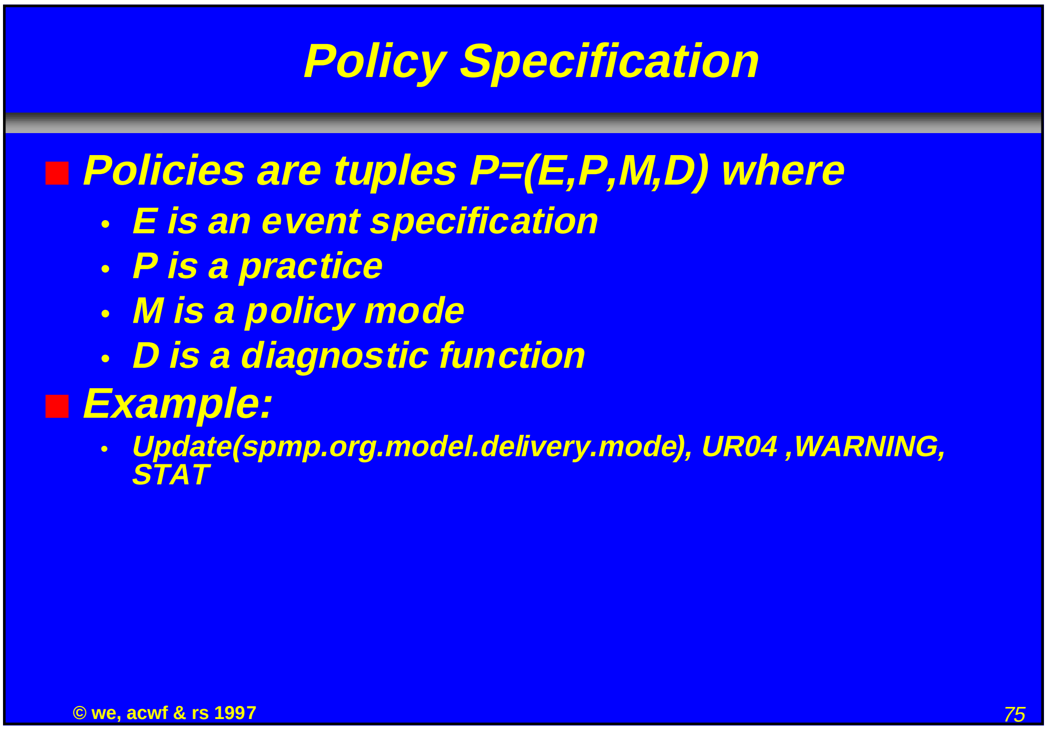
  $\forall r \in$ *urd.reqs.capabilities.fr:*
  (spmp.org.model.delivery.mode=incremental) $\Rightarrow$
  r.priority $\neq$ *0*

- Practices are conjunctions of properties
- Composite practices are conjunctions of component practices.



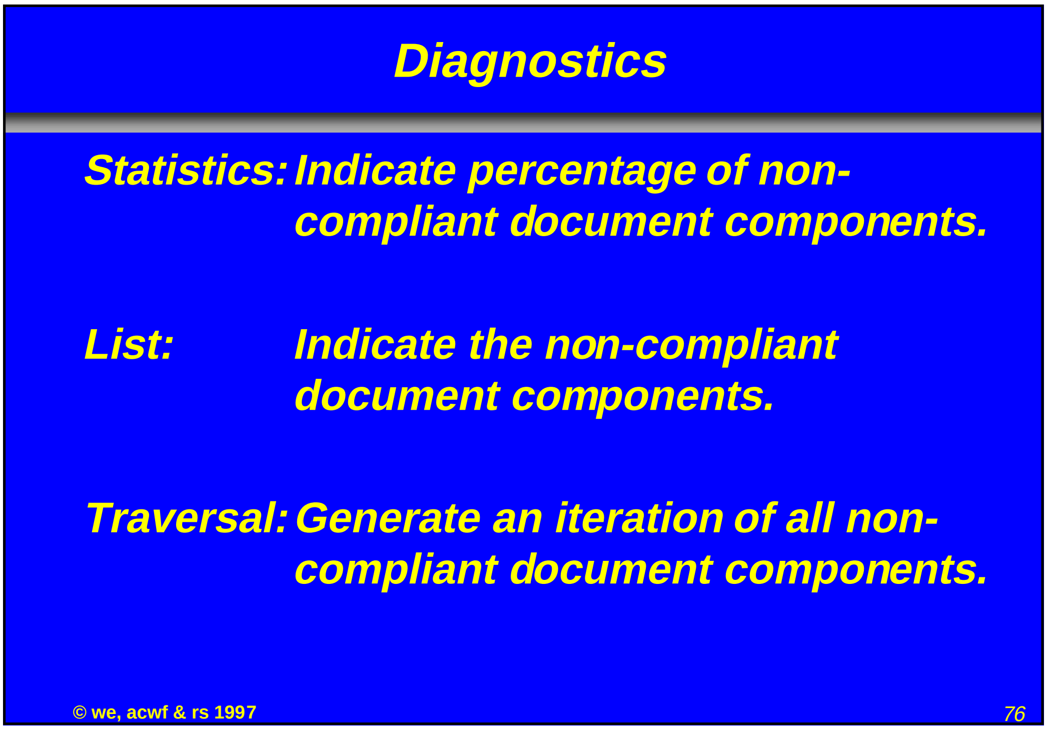
Event Specification

- Atomic Events:
  - Update
  - Close
  - Open
  - Baseline
- Logical Event Composition:
  - Open(doc) OR Update(att)
- Temporal Event Composition (as in FLEA):
  - Open(doc) THEN Update(att)
  - Update(att) IN-TIME(5h) Baseline(doc)
  - Open(doc) TOO-LATE(5h) Close(doc)



Policy Specification

- Policies are tuples P=(E,P,M,D) where
  - E is an event specification
  - P is a practice
  - M is a policy mode
  - D is a diagnostic function
- Example:
  - Update(spmp.org.model.delivery.mode), UR04 ,WARNING, STAT

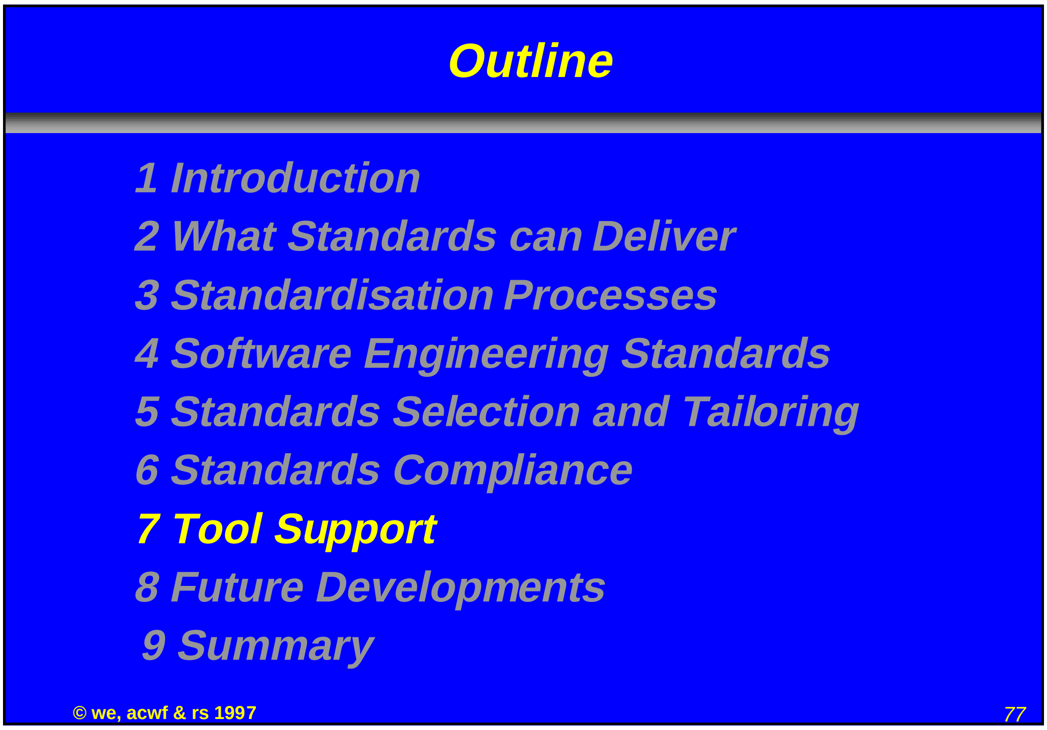


Diagnostics

Statistics: Indicate percentage of non-compliant document components.

List: Indicate the non-compliant document components.

Traversal: Generate an iteration of all non-compliant document components.



Outline

1 Introduction

2 What Standards can Deliver

3 Standardisation Processes

4 Software Engineering Standards

5 Standards Selection and Tailoring

6 Standards Compliance

7 Tool Support

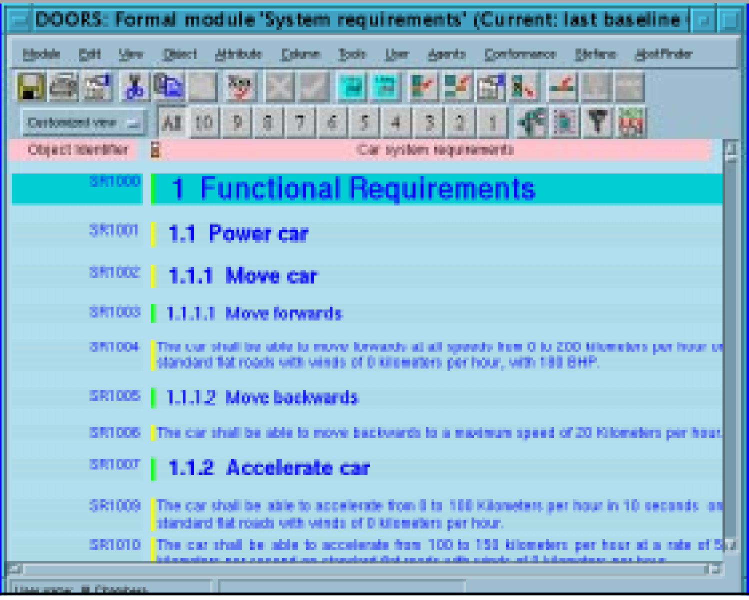
8 Future Developments

9 Summary

System Development Environments

- Select Document Management Tools
  - DOORS
  - MS-Office (Word + Excel + Powerpoint)
  - Framemaker
- Customise using
  - Scripting languages (DXL, Visual Basic)
  - Published component interfaces
- Example: DOORS

Document Management



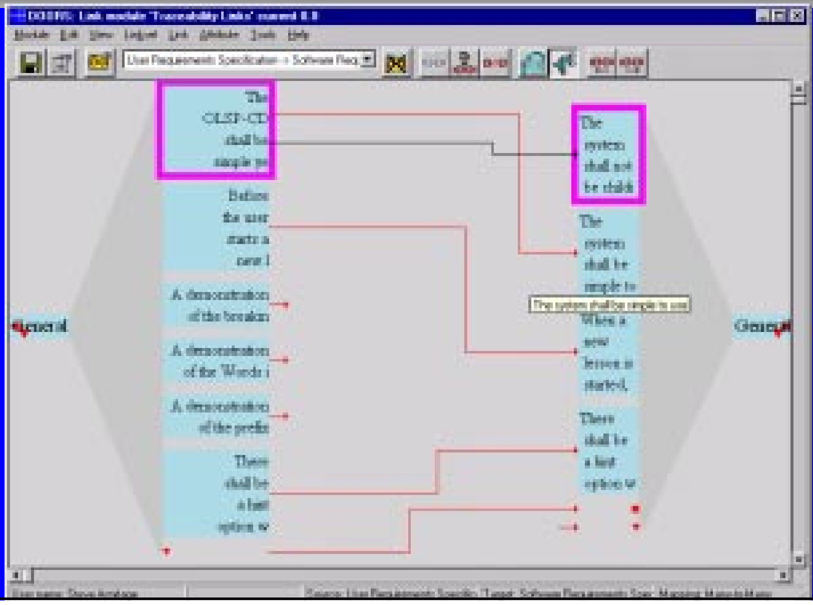
Module Edit View Object Attribute Column Tools User Agents Conformance Before HotFinder
Customized view
All 10 9 8 7 6 5 4 3 2 1

DOORS: Formal module 'System requirements' (Current: last baseline

Object identifier	Car system requirements
SR1000	1 Functional Requirements
SR1001	1.1 Power car
SR1002	1.1.1 Move car
SR1003	1.1.1.1 Move forwards
SR1004	The car shall be able to move forwards at all speeds from 0 to 200 kilometers per hour on standard flat roads with winds of 0 kilometers per hour, with 180 BHP.
SR1005	1.1.1.2 Move backwards
SR1006	The car shall be able to move backwards to a maximum speed of 20 Kilometers per hour.
SR1007	1.1.2 Accelerate car
SR1009	The car shall be able to accelerate from 0 to 100 Kilometers per hour in 10 seconds on standard flat roads with winds of 0 kilometers per hour.
SR1010	The car shall be able to accelerate from 100 to 150 kilometers per hour at a rate of 5 kilometers per second on standard flat roads with winds of 0 kilometers per hour.

User name: B Chambers

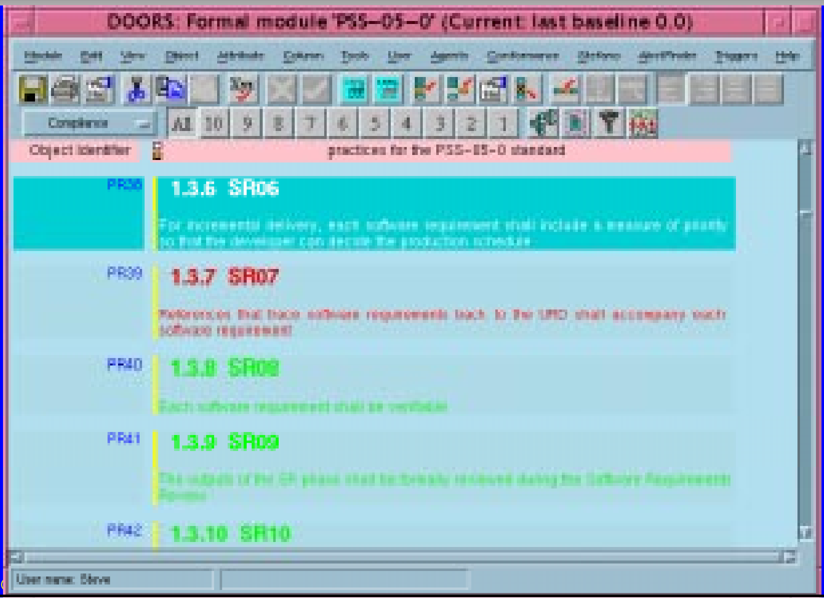
Traceability



DOORS: Link module 'Traceability Links' current 0.0
Module Edit View Linkset Link Attribute Tools Help
User Requirements Specification -> Software Req
The OLSP-CD shall be simple yo
Before the user starts a new l
A demonstration of the location
A demonstration of the Words i
A demonstration of the prefix
There shall be a hint option w
General
The system shall not be childi
The system shall be simple to
The system shall be simple to use
When a new lesson is started,
There shall be a hint option w
General

User name: Steve Armitage | Source: User Requirements Specific. Target: Software Requirements Spec. Mapping: Many to Many

Compliance Management



DOORS: Formal module 'PSS-05-0' (Current: last baseline 0.0)

Module Edit View Object Attribute Column Tools User Agents Conformance Stefano AlertFinder Triggers Help
Compliance
All 10 9 8 7 6 5 4 3 2 1
Object Identifier
User name: Steve

practices for the PSS-05-0 standard

PR38 **1.3.6 SR06**

For incremental delivery, each software requirement shall include a measure of priority so that the developer can decide the production schedule

PR39 **1.3.7 SR07**

References that trace software requirements back to the URD shall accompany each software requirement

PR40 **1.3.8 SR08**

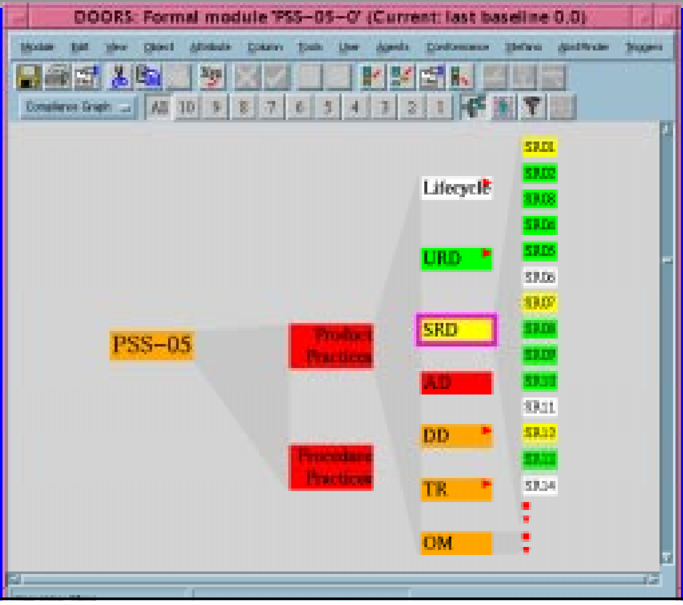
Each software requirement shall be verifiable

PR41 **1.3.9 SR09**

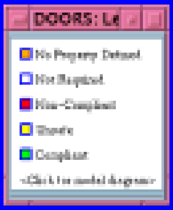
The outputs of the SR phase shall be formally reviewed during the Software Requirements Review

PR42 **1.3.10 SR10**

Compliance Overview



DOORS: Formal module 'PSS-05-0' (Current: last baseline 0.0)



DOORS: Le

🟧 No Property Defined

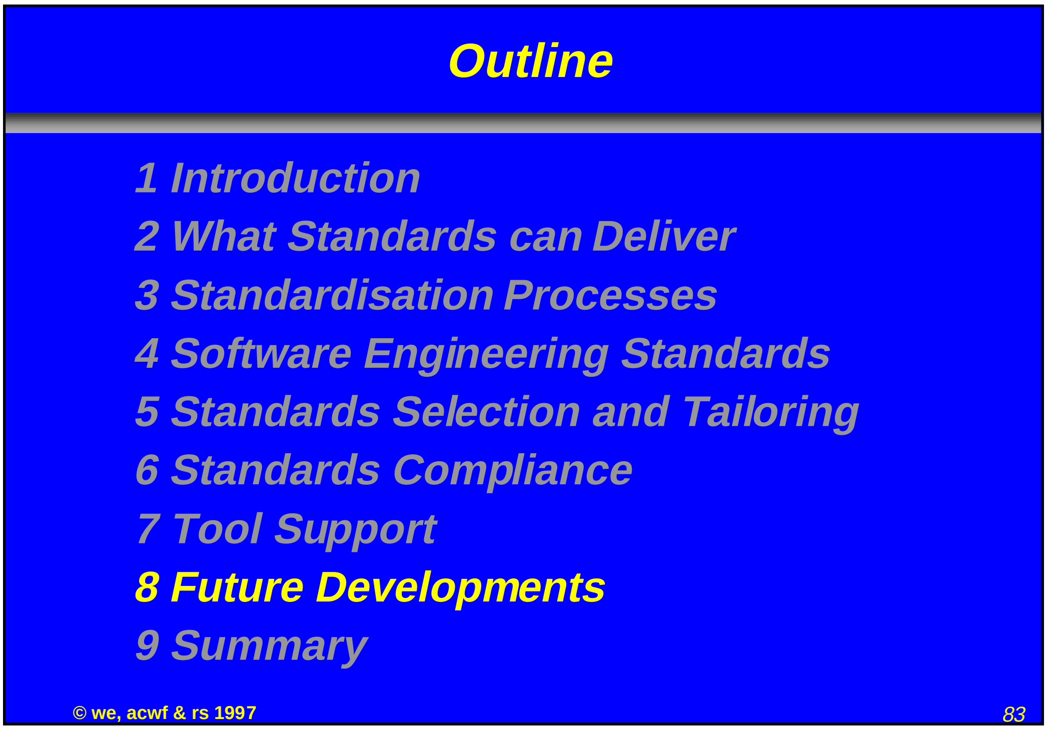
□ Not Required

🟥 Non-Compliant

🟨 Partly

🟩 Compliant

<Click for modal diagram>

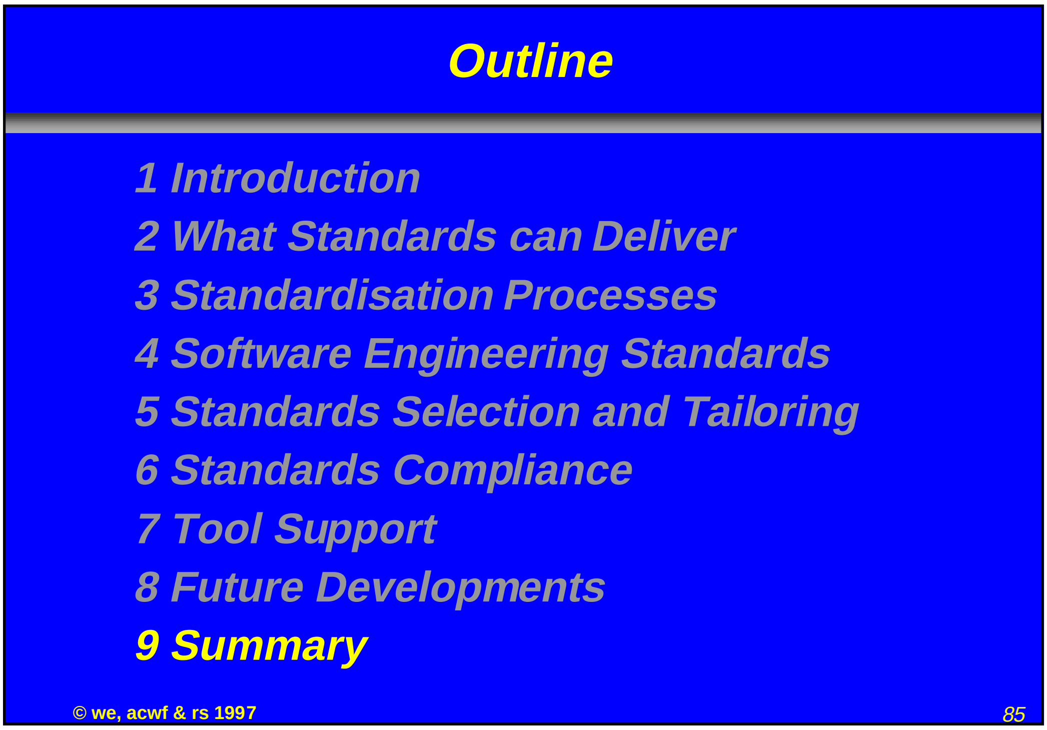


Outline

1 Introduction
2 What Standards can Deliver
3 Standardisation Processes
4 Software Engineering Standards
5 Standards Selection and Tailoring
6 Standards Compliance
7 Tool Support
8 Future Developments
9 Summary

ISO-15288

- ISO Standard for “Systems Engineering Lifecycle Processes”
- Extends ISO-12207 to system engineering processes
- Reflects composition of systems from systems, where each system has its own lifecycle.
- To be completed by 2000/01



Outline

1 Introduction

2 What Standards can Deliver

3 Standardisation Processes

4 Software Engineering Standards

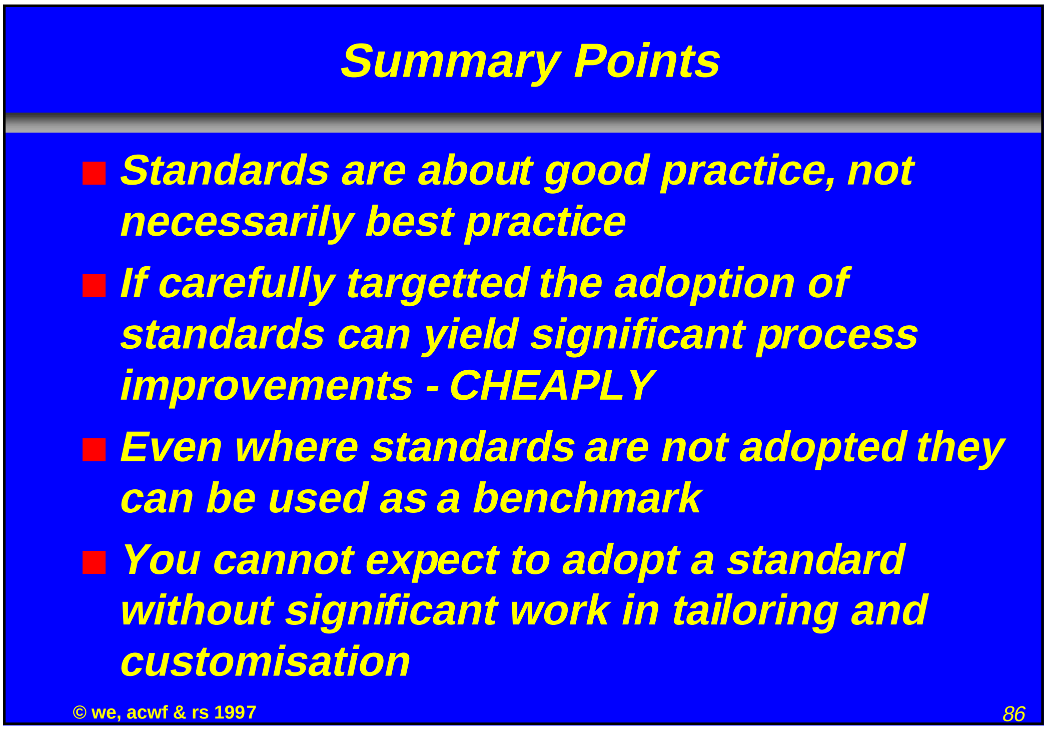
5 Standards Selection and Tailoring

6 Standards Compliance

7 Tool Support

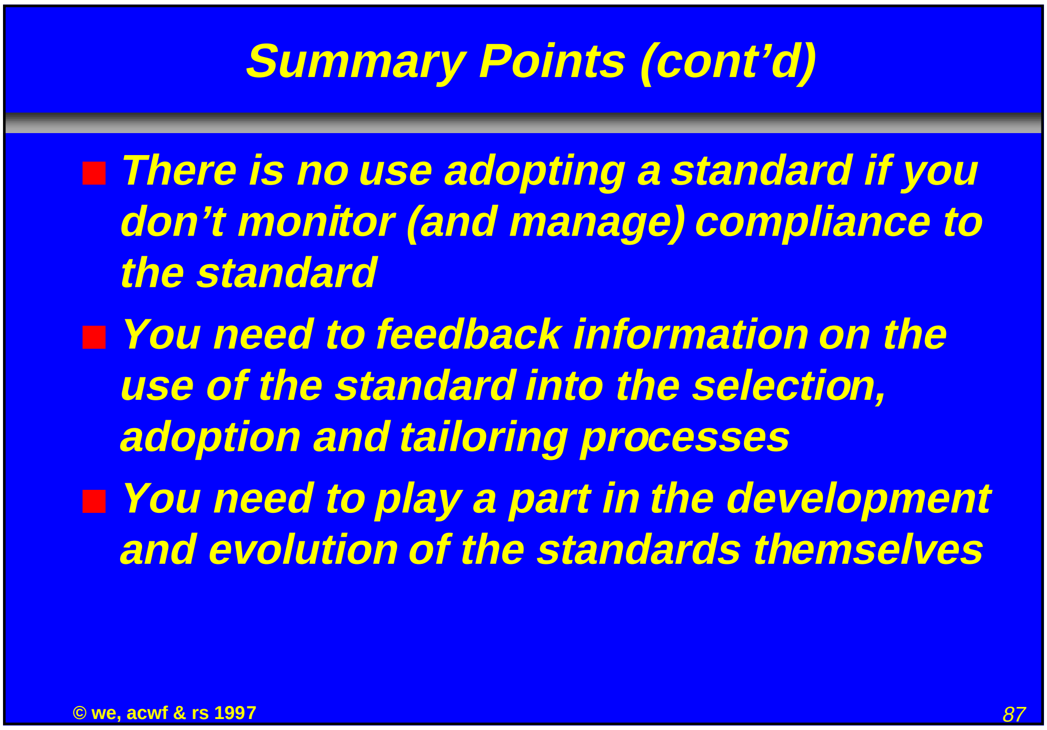
8 Future Developments

9 Summary



Summary Points

- Standards are about good practice, not necessarily best practice
- If carefully targetted the adoption of standards can yield significant process improvements - CHEAPLY
- Even where standards are not adopted they can be used as a benchmark
- You cannot expect to adopt a standard without significant work in tailoring and customisation



Summary Points (cont'd)

- There is no use adopting a standard if you don't monitor (and manage) compliance to the standard
- You need to feedback information on the use of the standard into the selection, adoption and tailoring processes
- You need to play a part in the development and evolution of the standards themselves

References

- Emmerich, W., Finkelstein, A., Montangero, C., Antonelli, S., Armitage, A., Stevens, R. (1997). Managing Standards Compliance. Submitted for Publication.
- Institute of Electrical and Electronical Engineers. (1995) IEEE Standard for Developing Software Life Cycle Processes. 1074-1995. IEEE Computer Society Press.
- Institute of Electrical and Electronical Engineers. (1997) Software Engineering Standards Collection. IEEE Computer Society Press.
- ISO/IEC (1994). Quality Management and Quality Assurance Standards - Part 3: Guidelines for the application of ISO 9001 to the development supply and maintenance of software. ISO 9000-3.
- Kuvaja, P.J., Simila, J., Krzanik, L., Bicego, A., Saukkonen, S. and Koch, G (1994). Software Process Assessment and Improvement - The Bootstrap Approach. Blackwell, Oxford.
- Mazza, C., Fairclough, J., Melton, B., De Pablo, D., Scheffer, A. and Stevens, R. (1994). Software Engineering Standards. Prentice Hall.
- Mazza, C., Fairclough, J., Melton, B., De Pablo, D., Scheffer, A., Stevens, R., Jones, M. and Alvisi, A. (1994) Software Engineering Guides. Prentice Hall.