Social Analysis and Dependable Systems Design

Prof. Ian Sommerville
School of Computer Science
St Andrews University
Scotland

St Andrews

- Small Scottish town, on the north-east coast of the UK
- Home of golf
- Scotland's oldest university (founded in 1413)
- Small university focusing on research and teaching excellence

Structure of talk

- Part 1: Background and rationale
  - Why social factors are important in designing dependable socio-technical systems
- Part 2: Integrating social analysis and software engineering
  - An example of one approach we have developed for using social analysis in software engineering

Background

- Software dependability
- The DIRC project - dependability of socio-technical systems
- Socio-technical systems
  - Cooperation
  - Awareness
  - Workarounds
- Social analysis
  - Ethnography

System dependability

- Availability
  - Will the system deliver service when required?
- Reliability
  - Will the system behave according to its specification?
- Safety
  - Could the system damage its users or its environment?
- Integrity
  - Will the system protect itself and its data from damage?
- Confidentiality
  - Will the system maintain ensure that access to data and resources is only permitted to authorised users?

Dependability research

- Body of research since the 1980s focusing on how to measure and improve software dependability
- Improving software dependability
  - Fault avoidance. Use methods and techniques during system development that avoid introducing faults into the software or that detect faults before the software is deployed.
  - Fault tolerance. Use run-time support for software that detects system faults before these result in system failure and initiates actions to recover from these faults.

Dependability engineering

- Technical advances to improve software dependability have mostly been applied in control systems or in systems where continuous availability is required.
- These have been largely successful
  - It is certainly possible to create software systems that exhibit very high levels of availability (e.g. telephone switching software) and/or very low failure rates (e.g. flight control systems)
  - However, all of these depend on a very detailed software specification with dependability defined with respect to that specification
  - The techniques also assume an approach to software engineering where conventional programming languages are used for software development

Dependability problems

- There remains a significant level of failure in the broader socio-technical systems where software systems are used (e.g. medical information systems)
- Some of the failures that occur are a consequence of the software specification failing to recognise the practical realities of the environment where the system is used
- Dependability engineering is very expensive and is only really justifiable in the most critical systems

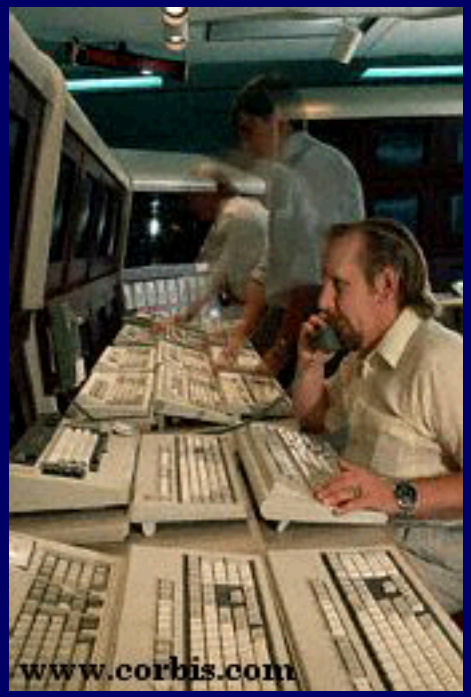
The DIRC project

- Research project focusing on the dependability of socio-technical systems rather than software.
- Socio-technical systems
  - Hardware, software, processes, organisations, people
  - Hospital information system, air traffic control system
- DIRC assumption
  - Dependability should be defined w.r.t the socio-technical system which includes the software NOT the software specification

DIRC goals

- Derive methods and techniques to improve the overall dependability of socio-technical systems
  - Examine the broader socio-technical systems that use software to help understand how these are affected by software dependability/undependability
  - Look at modern approaches to software development to understand how dependability issues are considered
  - Derive methods, techniques and tools to help improve the fit between deployed software and the broader socio-technical system

Socio-technical systems



- Computer-based systems are part of broader socio-technical systems that include the technical system, processes, people and organisational procedures
  - Air traffic control
  - Medical imaging
- Socio-technical systems are inherently cooperative systems involving both synchronous and asynchronous cooperation

System dependability

- Dependability isn't about conforming to a specification but rather reflects the system's ability to cope with human failures, unusual and unexpected circumstances and the changing requirements of stakeholders
- Work practice evolves to deliver dependability
- Dependability is achieved through
  - Cooperation
  - Awareness
  - Workarounds

Dependability foundations

- Redundancy
  - Ensure that there is spare capacity in the system that can be brought into use in the event of system failure
- Diversity
  - Ensure that there are different ways in the system to achieve the same goal
- Inherent tension between efficiency (use of resources) and dependability
  - Automation may lead to better utilisation of resources but sometimes reduces the redundancy and diversity in systems. It can reduce the overall dependability of systems

Working practices

- Responsive and reactive
  - People change their working practices in response to new information and they react rapidly to unusual circumstances
- Inherently flexible
  - If documented procedures and processes exist, they are often interpreted in different ways by different people and may be subverted in subtle but important ways
- Professional
  - Most people adopt a professional attitude to their work and design the work to take into account their professional skills
- Hard to articulate
  - It is difficult for practitioners to articulate the essential features of everyday tasks

Rule-based cooperation

- Some processes are explicitly cooperative and involve different people working on the same artefacts at different times
- These are the types of process that may be automated using workflow systems and specified using process models. There is a defined sequence of operations required and a division of work across these operations
- In some cases, process fragments can be enacted by automated services
- Generally, rule-based approaches can only codify how to react to a limited number of exceptions.

On-demand cooperation

- Knowledge-based processes may have elements of pre-defined cooperation but more cooperation is 'on-demand' i.e. people cooperate when they need to do so. The patterns of cooperation and its synchronicity are impossible to specify in advance
- The division of labour is flexible and is constantly renegotiated, often implicitly based on the current demands of the work
- Cooperation depends on the knowledge of the agents involved. It cannot be defined in advance

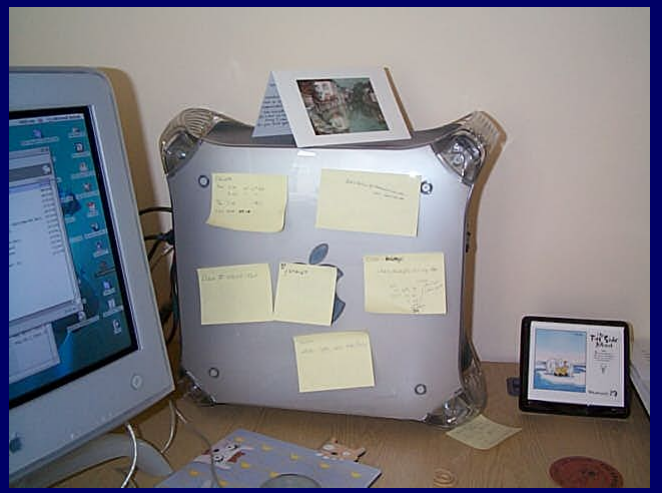
On-demand cooperation

- On-demand cooperation is an informal process.
  - Documents are passed from A to B with scribbled notes in the margin giving information about what has been done and what is required
  - People leave notes for themselves and others about actions and artefacts
  - Informal meetings are recorded by annotating documents with the conclusions of these meetings
- On-demand cooperation is the principal mechanism for exception management in many processes
  - When things go wrong, the formal process models are often discarded and opportunistic, on-demand cooperation is used to handle the exceptions

Awareness

- Work often depends on the awareness of what others people are available, what they are doing and what they have done
- An informal notion - formalising awareness changes its nature and is practically impossible
- Workplaces are often arranged to support awareness
  - Public and private spaces
  - Co-location of related tasks
- Awareness may be a trigger for on-demand cooperation

Office reality



Awareness and reminders

- Informal mechanisms of communication
  - Universal - no previous knowledge is required to use them and they may be used anywhere
  - Visible - they are obvious on a document or in a workplace
  - Identifiable - different handwriting identifies the producer. In some cases, explicit actions (different colours of pen) may be used to identify the writer.
- Mechanisms for awareness
  - People use stickies for reminders of what to do
  - Others can look at these stickies to become aware of what is being done

Flight strips

- The artefact as an audit trail
  - Flight strips are used to record ATC commands
  - The written annotations on the strip are visible to all of the team and provide awareness of the state of the aircraft

13.05	POL	350 ↑	BA5612 5466 M/B737/C T420	131.05 80 EGLL W22 UB4 B22 EGPD	TRENT 12.58 350

Awareness and dependability

- Strips may be 'cocked out' of the rack showing that a flight is, in some way, a special case
- Different people can write on the strips using different colours of pen - awareness of who has done what
- The number of strips on the rack provides workload awareness - how busy is a sector and what workload adjacent sectors should plan for

Workarounds

- Workarounds are deviations from some 'normal' process that people invent to cope with problems
- Workarounds allow the work to be done in situations where information or other resources are not available
- Workarounds often involve 'breaking the rules', individuals exceeding their authority or taking on new roles
- Although often not formally sanctioned in an organisation, they are generally known and tolerated as they lead to enhanced dependability

Workarounds are important

- In a chemotherapy unit, a common failure was that the doctor involved forgot to order the required drugs for the patient's treatment
- The workaround for this was for nurses and the hospital pharmacy to 'break the rules'
- Nurses wrote the prescription and the pharmacy dispensed the drugs. The doctor then signed the prescription when he or she arrived for the treatment
- Then the hospital introduced a computerised prescribing system which automatically sent prescriptions to the pharmacy
  - This system, of course, embedded the rules and only doctors were authorised to write prescriptions!

Understanding informality

- The details of some tasks, particularly those which are context-sensitive are difficult to articulate. Observing people doing these tasks is a better way of understanding the work than asking them about them.
- Ethnography is an observational method of social analysis whereby a social scientist becomes absorbed into a culture and observes the details of the practices in that culture.
- Its fundamental assumption is that details are as important as abstractions and details can only be discerned by prolonged observation
- It can be used, in a modified way, to study various types of work, particularly where this work has a social element

Benefits of ethnography

- *Understanding the real process*
  - Whatever process is specified, practitioners rarely follow the formal process. Providing process support based on this formal process has been, in many cases, unsuccessful
- *Understanding cooperation*
  - Many tasks are explicitly or implicitly cooperative. As ethnography is a method of social analysis, it can help understand this cooperation. Structured analysis methods and task analysis tend to factor out cooperation from the process
- *Understanding awareness*
  - In some types of work, actions depend on awareness of other actions. Ethnography, with its focus on detail, can recognise this.

Problems with ethnography

- *Non-judgmental*
  - The ethnographer presents information about the work without making an assessment of its importance
- *Prolonged*
  - Ethnography (typically) takes a long time
- *Personalised*
  - Ethnographers keep detailed notes of their observations but our experience is that these notes are not readily understood by anyone apart from the observer
- *Disassociated*
  - Up till now, ethnography has been a separate part of the analysis process. There has been little work on using ethnography with other forms of analysis

Questions?

Socio-technical systems engineering

- Taking an holistic view of systems engineering where we consider human, social and organisational as well as technical issues in the system design.
- Moving from ethnography to an approach to social analysis that can be more readily integrated with systems engineering processes.
- Making observational techniques from the social sciences accessible to systems and software engineers.

Social analysis and software engineering

- Evolving ethnography for use in different settings
  - Ethnographic viewpoints
  - Cultural probes
- Integrating social analysis into systems engineering processes
  - The Coherence method
  - Responsibility modelling
- Generalising ethnography
  - Patterns of interaction
- Ethnography and software testing

Ethnography in systems engineering

- There is a mismatch between the representations used by ethnographers (free text notes, photographs, recordings) and those used in systems and software engineering methods
- Few systems engineering projects have the resources to employ ethnographers
- Ethnographers have no tradition of generalisation with the consequence that organisational learning is difficult

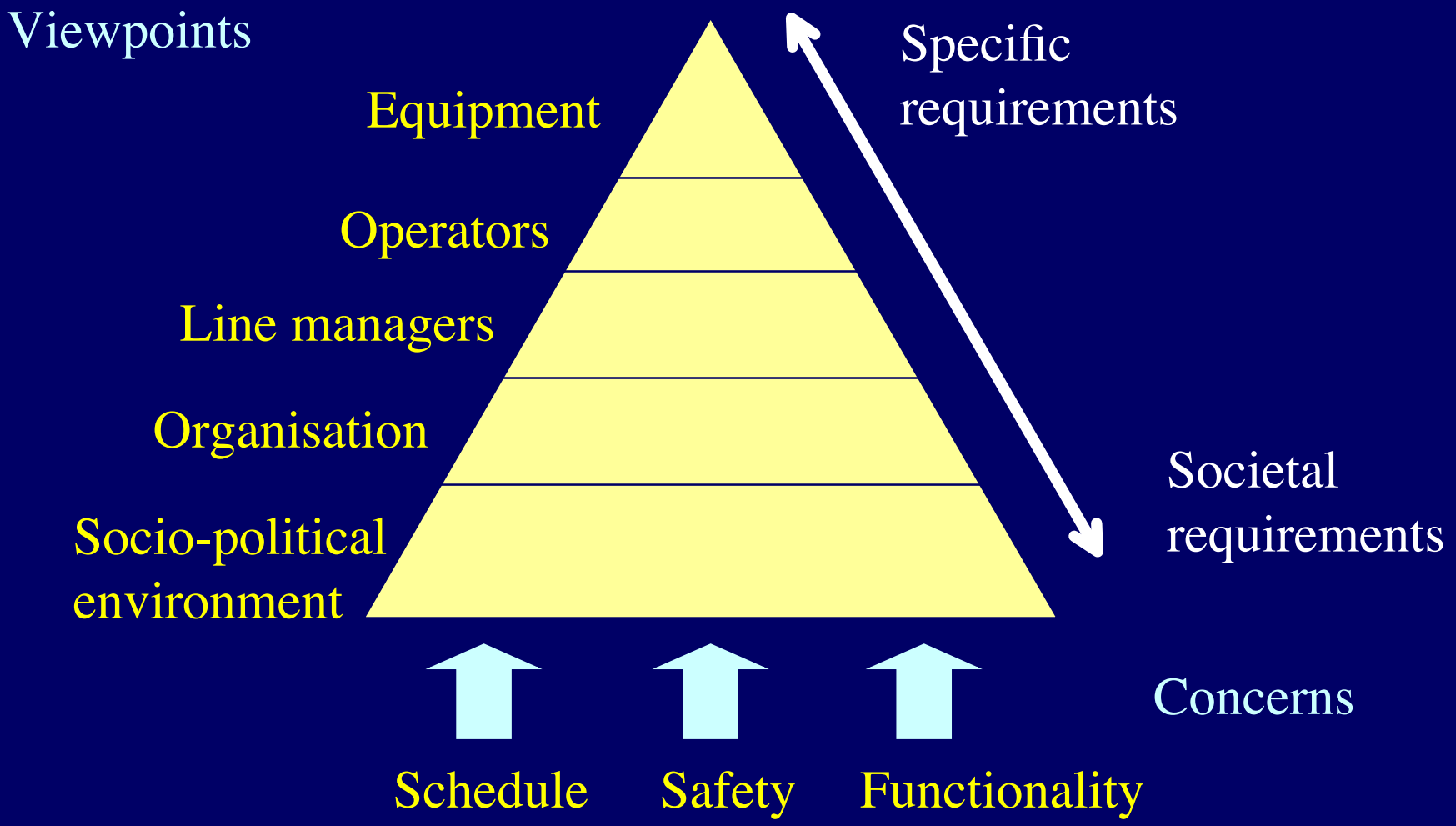
Coherence

- A 'lightweight' method which allows requirements engineers to apply some of the lessons we have learnt from several years of ethnographic studies
- The method includes
  - Process guidance - how to look for and recognise social issues which may affect the requirements for a system
  - Representation guidance - how to represent the social analysis using graphical system models
- Notations in Coherence are based on the UML
  - Accepted standard for OO analysis
  - Good quality tool support is available (with some extensibility)

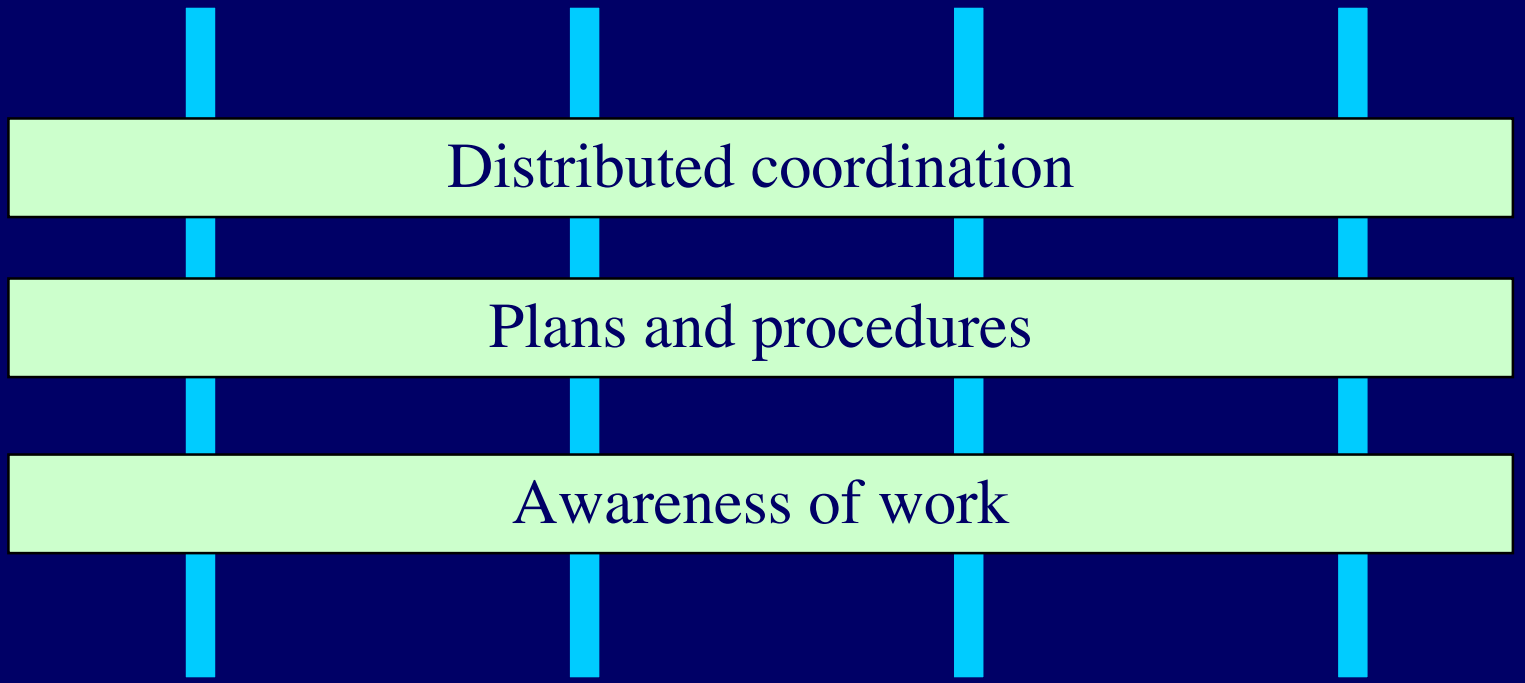
Viewpoints and concerns

- Viewpoints
  - Perspectives on a process or system which provide a partial description of the system. The descriptions may represent the existing process or system or the desired process or system. They are a means of *organising and structuring* the elicitation and presentation of system requirements
- Concerns
  - Issues which are of relevance to all viewpoints and which are orthogonal to them. In requirements analysis, these may represent business goals such as 'time to market' or overall system attributes such as efficiency, safety and functionality.

Viewpoints and concerns



Social viewpoints and concerns



Social viewpoints

- Social viewpoints give requirements engineers guidance on how to organise their social analysis
- We have identified three viewpoints that seem to be fairly universal
  - Distributed coordination
    - The coordination of people and tasks as part of everyday work
  - Plans and procedures
    - The role of organisational plans and procedures which both facilitate and inhibit processes
  - Awareness of work
    - The organisation of activities to promote awareness of the work by the people involved in the process

Viewpoint examples

- Distributed coordination
  - Air traffic control is a team activity involving 5 controllers in each sector. How do they share tasks, cope with heavy loads, coordinate their activities etc.
- Plans and procedures
  - In an ATC system, different teams have evolved different control strategies which follow to a greater or lesser extent the formal ATC procedures
- Awareness of work
  - Awareness of other controller activities is critical for safety in an ATC system. It is also important for workload planning

Concerns

- Paperwork and computer work
  - How is paper and technology used in the workplace?
- Skill and the use of local knowledge
  - How are skills and local knowledge applied?
- Spatial and temporal organisation
  - How does the physical and temporal organisation affect the performance of the work?
- Organisational memory
  - How is implicit organisation knowledge used to facilitate the performance of work?

Paperwork and computer work

- Distributed coordination
  - How is work coordinated through the use of paper and computer-based forms?
  - How do forms embody the work and the people doing the work?
- Plans and procedures
  - To what extent do people trust descriptions of the system that they use?
  - If a procedure specifies the use of specific representations, is this use monitored by the organisation?
- Awareness of work
  - How does paper and the affordances it offers facilitate awareness

Process steps

- Determine the appropriateness of concerns in the current content
- Elaborate concerns to more specific questions
- Identify additional viewpoints (not social viewpoints) in addition to the social viewpoints
- Interact with stakeholders to understand the system requirements
- Elaborate requirements as annotated use-cases and supporting UML models

Concern choice

- Decide whether or not the identified social concerns are relevant in a particular context
  - For example, the spatial organisation concern is likely to be important where work is co-located and synchronous but less significant where work is distributed and asynchronous
- Identify other concerns which are relevant
  - Social analysis is part of the elicitation process but its coverage is incomplete. Other concerns e.g. based on business goals may also be relevant and these should be identified at this stage

Concern elaboration

- Concerns are elaborated to more specific concerns and, finally, into a set of questions. The analyst looks for the answers to these questions during the elicitation process
- Spatial and temporal organisation
  - Sub-concerns might be use of shared space, use of private space, physical workspace layout, synchronous organisation, asynchronous organisation
  - Possible questions:
    - How are shared workspaces organised?
    - Does data have a 'use-by' date
    - How does work move from shared to private workspaces
    - How does the physical layout of the workspace facilitate information retrieval

Viewpoint identification

- We have already identified 3 social viewpoints. This stage is concerned with identifying other viewpoints which may be relevant and understanding the relationships between these and the social viewpoints
  - End-user viewpoint - concerned with specific tasks
  - Management viewpoint - concerned with the results produced by end-user viewpoints
- Relationships with social viewpoints
  - End-user tasks may depend on distributed coordination
  - Plans and procedures may explicitly define end-user task processes
  - End-user tasks may be facilitated by awareness of other work

Requirements discovery

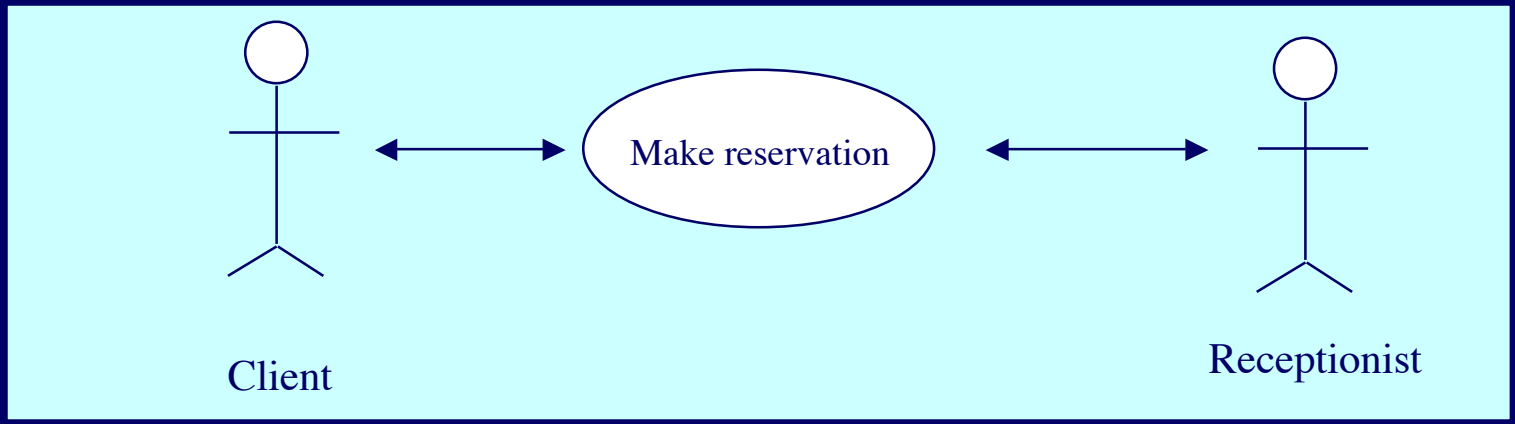
- Investigation of a workplace to develop a better understanding of that workplace. Requirements emerge from this understanding
- Driven by concerns not viewpoints. Concerns provide the questions that should be answered for each viewpoint. Social concerns may also be relevant to other (non-social) viewpoints
- Essentially opportunistic but facilitated by the questions which are generated from the concerns.
- Questions may be answered through interviews, observation, existing documentation, etc.

Awareness of work viewpoint

Name:	Awareness of work
Focus:	How the physical organization of the control suites affects how controllers can make sense of each other's activities. How controllers monitor the work of other controllers, and how controllers orient their work to facilitate others monitoring it.
Concerns:	Paperwork and computer work Skill & the use of local knowledge Spatial and temporal organization ~~Organizational memory~~ Safety Volume of traffic
Sources:	Controllers, and observation of controllers at work
Requirements:	AW1 (Making work available) AW2 (Availability of awareness information) AW3 (Relationship of suite layout to controlled airspace)

Object-oriented analysis

- Jacobsen's approach to OOA which is reflected in the UML is based on the notion of use-cases where a use-case represents some interaction with a system

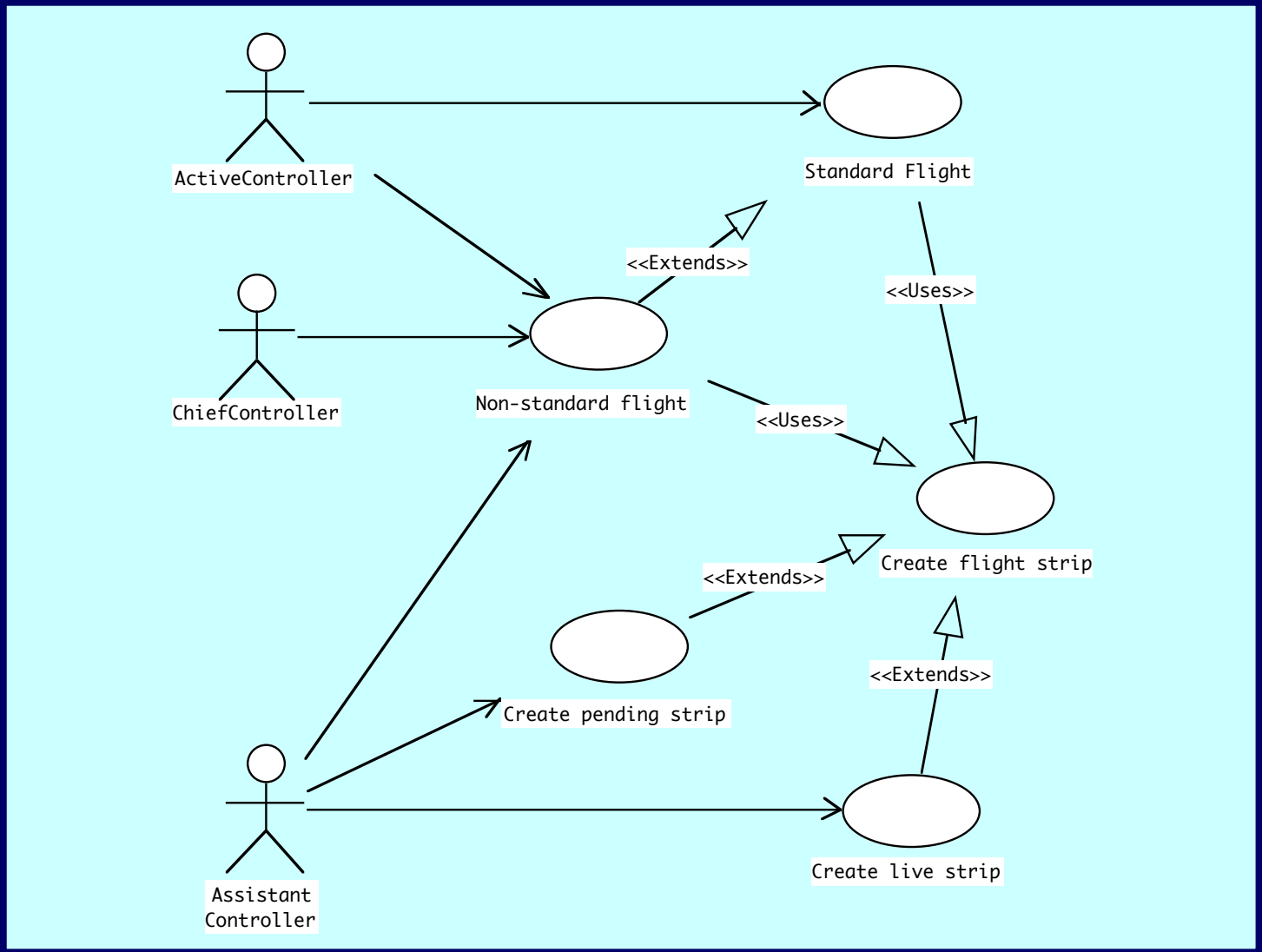


- Applying the Coherence approach helps us to find and understand relevant use-cases and helps with the documentation of use-cases

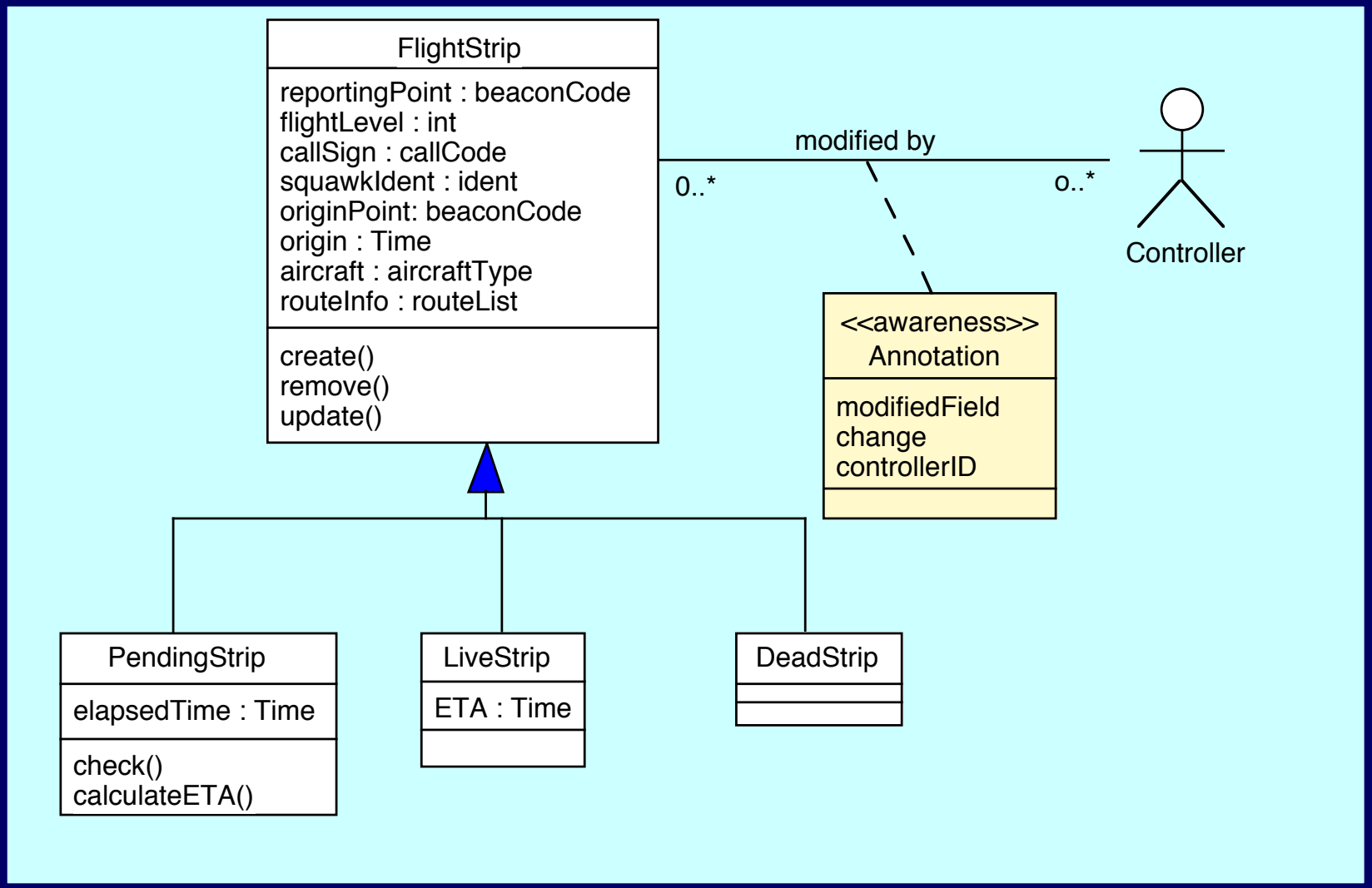
Coherence and use-case models

- **Actor** Interactor stakeholders are identified and then used to generate viewpoints
- **Use case** Use case descriptions are generated by plans & procedures viewpoint
- **Problem domain object model** Problem domain objects are identified by distributed coordination and awareness of work viewpoints
- **Object model** Fragments of model are generated by awareness of work viewpoint
- **Interface descriptions** Not directly addressed by Coherence, but can be recorded in UML models

Use-case identification



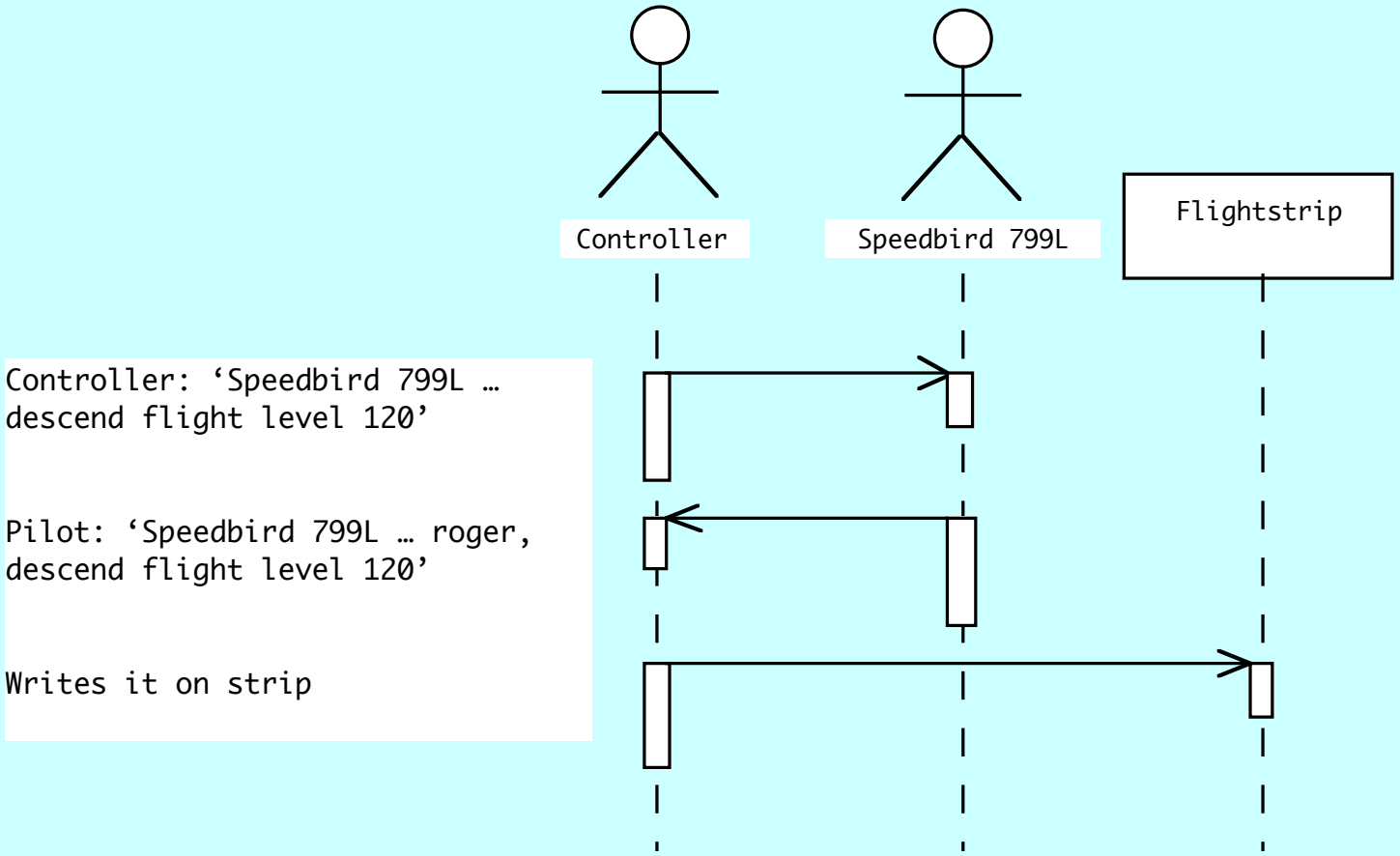
Awareness annotations



Modelling communication

- Providing system designers with models of communications between the participants in a process helps them develop an understanding of how to support that communication
- The distributed coordination viewpoint captures communications. These can be modelled using (extended) UML sequence diagrams that show interactions between people as well as interactions between a system end-user and the objects that are modified in that interaction

Flight coordination



Coherence benefits

- Provides a framework for social analysis that can be used by software and systems engineers
- Provides a means of structuring the presentation of fieldwork to engineers
- Uses structured notations that are accessible to engineers
- However, more work required on how to translate insights from social analysis into design recommendations

Integrated STSE

- New project (starting later this year) whose aim is to integrate this work with other approaches to create a socio-technical systems engineering process
- Integrated with work on design for failure and modelling responsibilities in complex systems
- Part of the UK's research programme in Large-Scale Complex IT systems (Southampton, St Andrews, York, Leeds, Oxford Universities, IBM, Rolls-Royce, NHS..)

Summary and conclusions

- System dependability is more important than software dependability
- Techniques for achieving software dependability do not translate well to modern business systems
- We must understand the socio-technical environment of a system so that we can design for failure
- Ethnography is effective but unrealistic for most businesses
- The COHERENCE approach uses socio-technical viewpoints as a means of integrating social analysis with system requirements engineering