Intelligent User Experience Engineering

Mark Neerincx

Task Analysis

Scenario-Based Design

Example: Adaptive Automation

Module 2

UX Engineering

- ***iterative*** process with active involvement of end-users
  - to better understand their *support needs*
  - to enhance *user acceptance*.
- ***collaborative*** process of multi-disciplinary team
  - to address the human factors, technical, statutory or legislative, and political issues.
- with general ***usability objectives***
  - *Effectiveness:* accuracy and completeness
  - *efficiency*, the resources required (e.g., time, mental effort)
  - *user's satisfaction* (user comfort and acceptability)
  - *learnability* (effectiveness, efficiency with minimal training)
- and additional ***experience objectives***
  - e.g., fun, complacency, engagement, ...

Scenario-Based Design

- Scenarios support reasoning about situations of use, even before those situations are actually created

Scenarios include

- a setting (starting state)
- agents or actors, each actor has goals
- a sequence of actions and events

- Scenarios can be elaborated as prototypes, through the use of storyboard (sequence of snapshots), video and rapid prototyping tools.

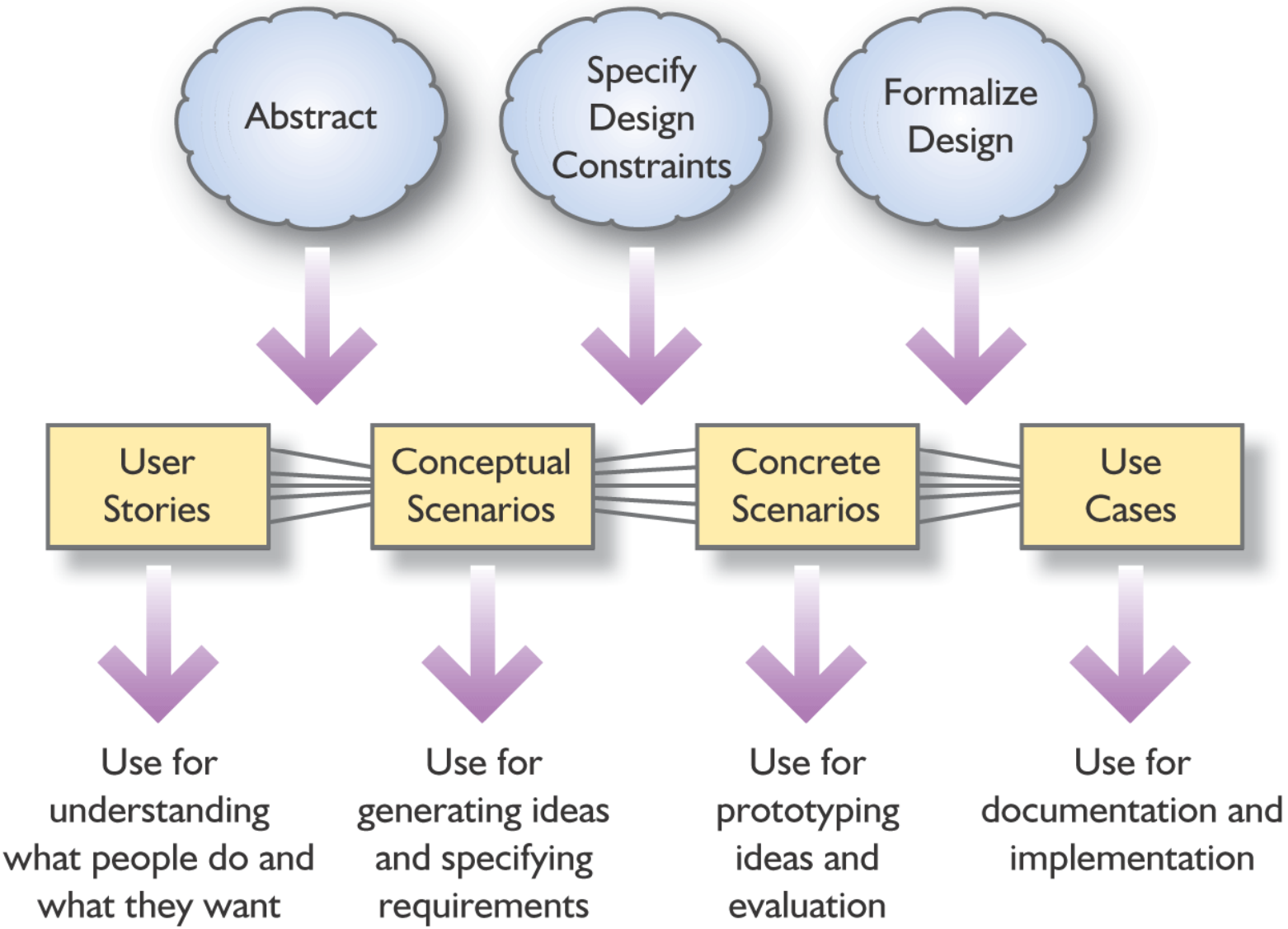


Figure 8.2 Scenarios throughout design.

Why Scenarios?

- evoke reflection
- are concrete and flexible
- afford multiple views of an interaction
- can be abstracted and categorized (“patterns”)
- improve accessibility of design activities by promoting work-oriented communication among stakeholders

Scenario Types, Refinement

(Start with root concept and context description)

Problem scenario

- a story about the problem domain as it exists prior to technology introduction

Design scenario

- a story that conveys a new vision
  - activity: narratives of typical or critical services
  - information: details on info provision
  - interaction: details of user interaction & feedback

⇒ Claims analysis (=> UX specification)

(iterative process)

Use Cases

- are intended to be a complete description of what a system will do
- a scenario is an instance of a use case (a scenario specifies functionality in the context of use and focuses less on completeness of coverage)

- link with Software Engineering

[UC_Nr]	Number used to link requirements and claims to use case.
Goal	What is achieved by carrying out the use case.
Actor	Main human (or possibly machine) actors.
Precondition	The state of the system or user just before using the function.
Post condition	The state of the system or user just after the function was used.
Trigger	Defines the event (e.g., time, alarm) when a user needs the functionality or how the system knows that the function needs to be carried out.
Main Success Scenario	A top-to-bottom description of an easy to understand and fairly typical action sequence in which the actor's goal is accomplished.
Alternative Scen	Other ways to succeed, and the handling of the most important failures
Claims	List of claim-numbers that link to this use case
Requirements	List of requirement-numbers that link to this use case

Task Analysis and Scenarios

- Specify task structure (e.g., via HTA)
- Define context
- Specify scenarios
- Check whether scenarios cover the context and task structure, if not, add/refine scenarios
- Derive requirements
- Perform claims analyses
- And test

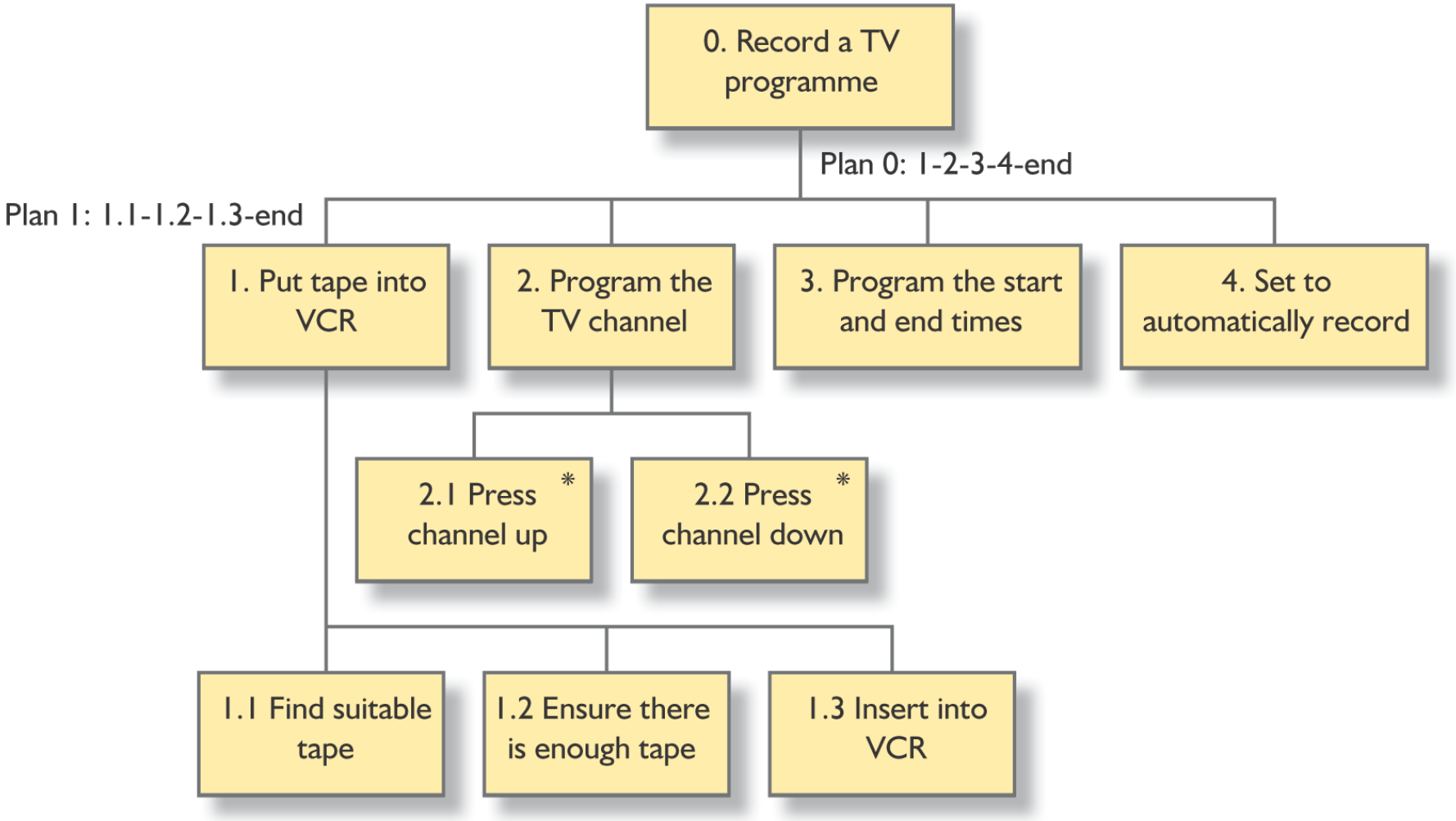


Figure 20.2 HTA for programming a VCR.

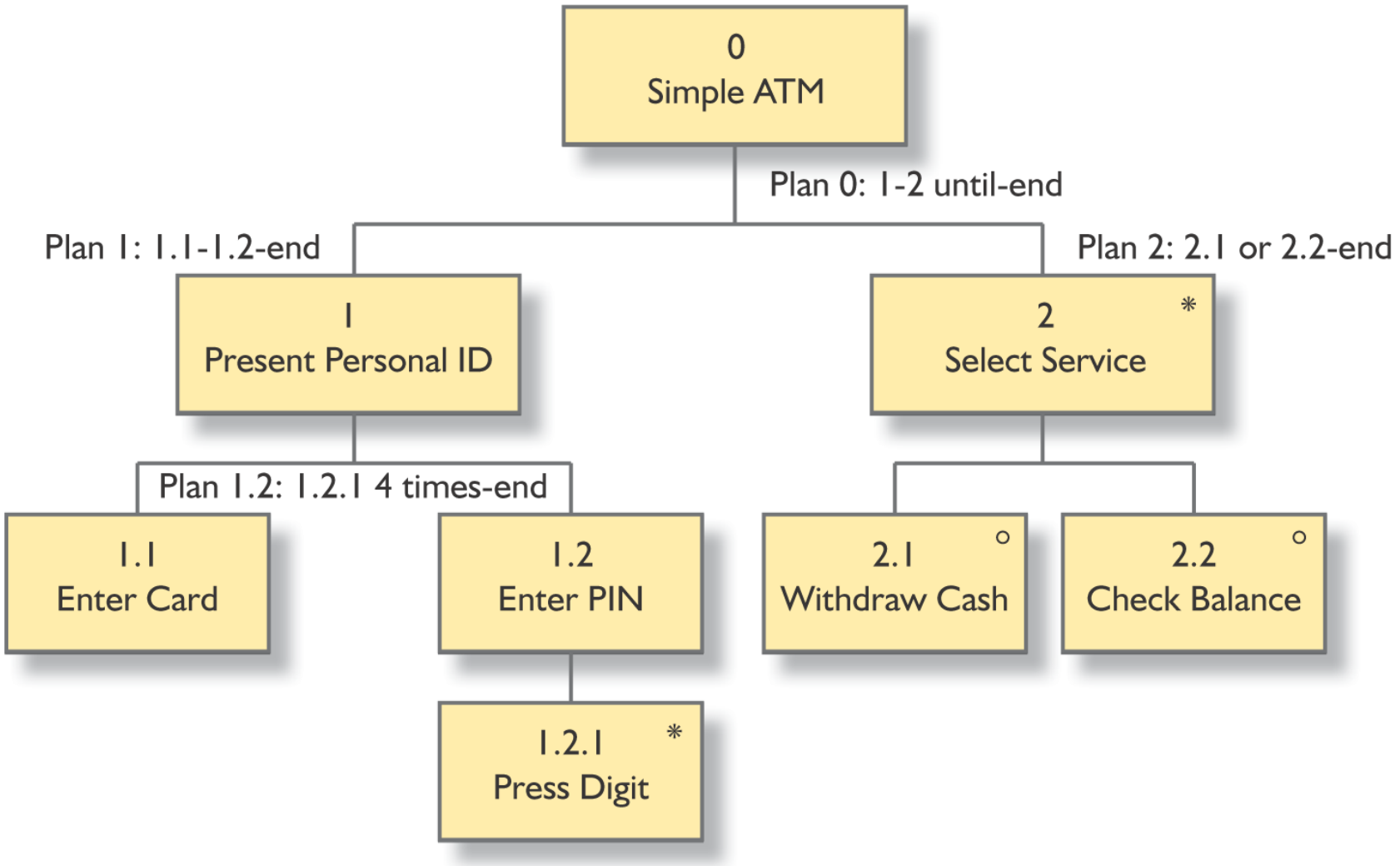


Figure 20.3 Hierarchical task model for a portion of an ATM.

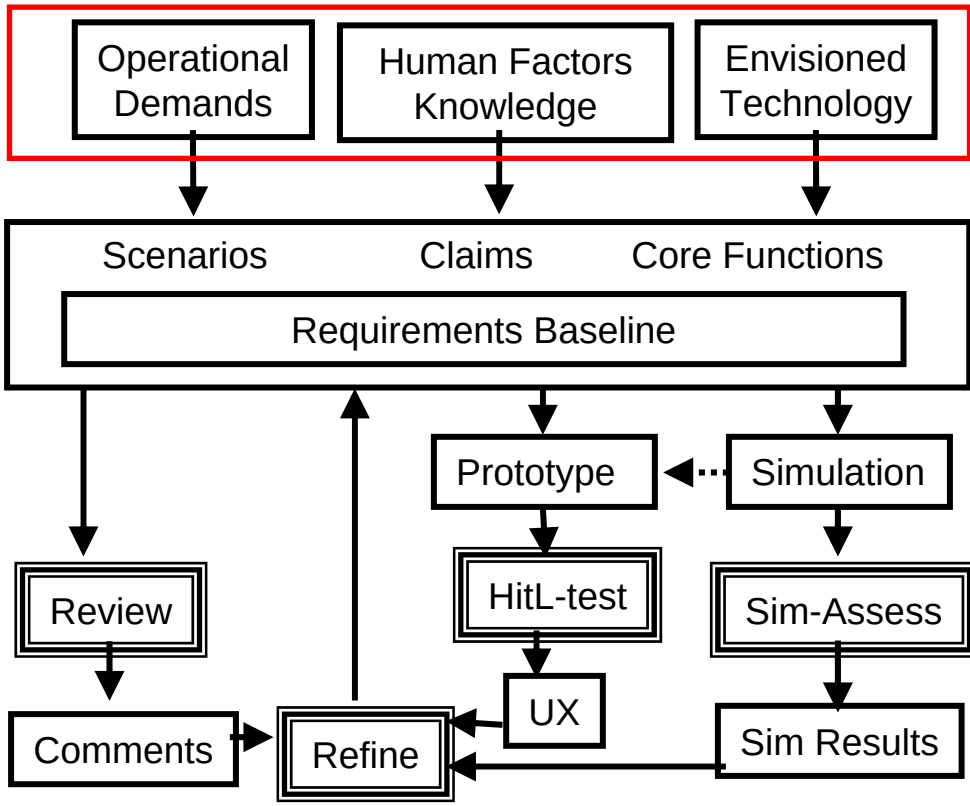
Adaptive Automation

Derive: Work Domain & Support analysis

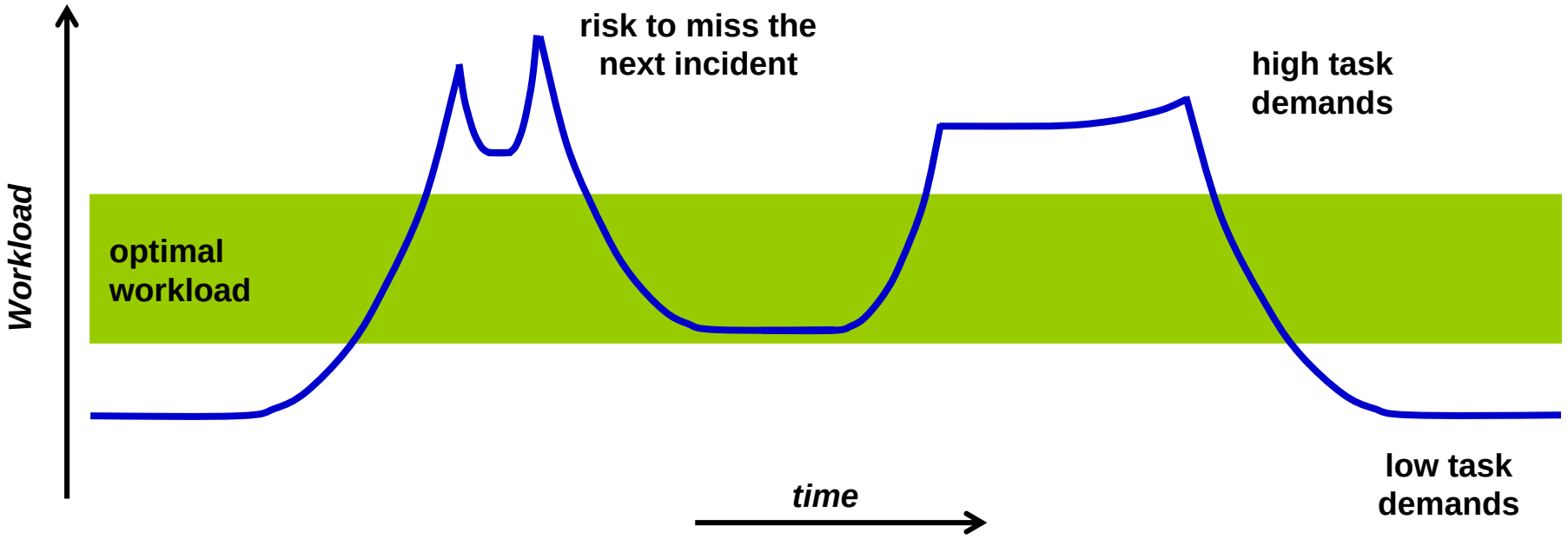
Operational demands
- Littoral waters
- Variable work demands
- Smaller crews

Human factors
- Human-in-the-Loop
- Workload
- Situation awareness
- ...

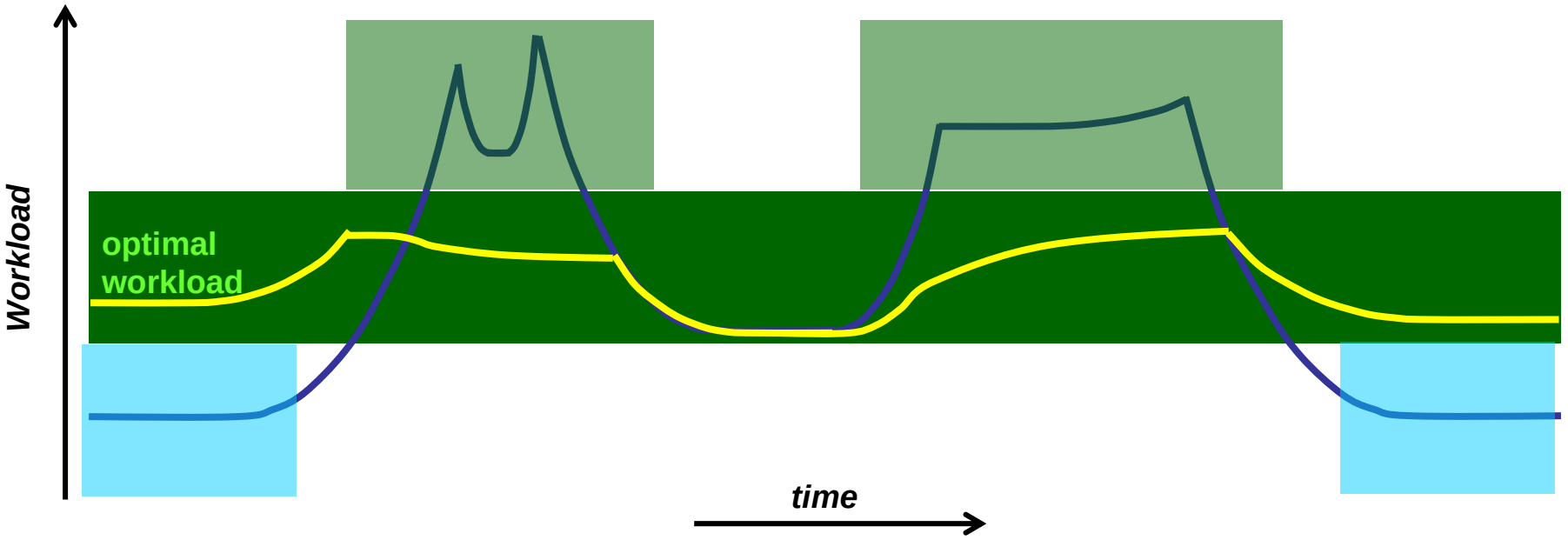
Technology
- Intelligent combat & platform management systems
- Adaptive automation
- Network-centric vs platform-centric



Variable Workload



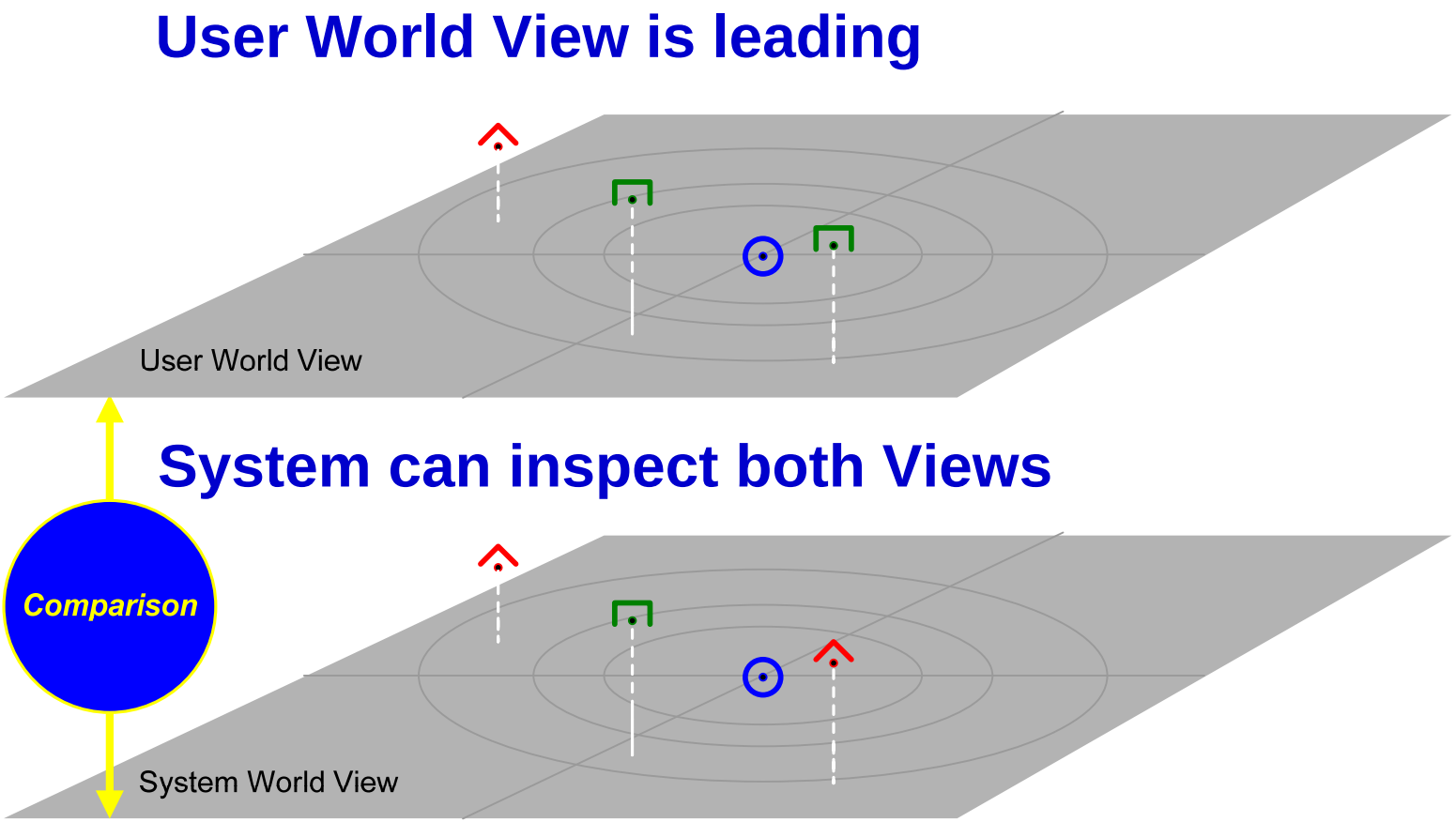
Root Concept: Variable Automation



Air Defense Task: Object identification

- Unknown
- Friendly
- Assumed friendly
- Neutral
- Suspect
- Hostile

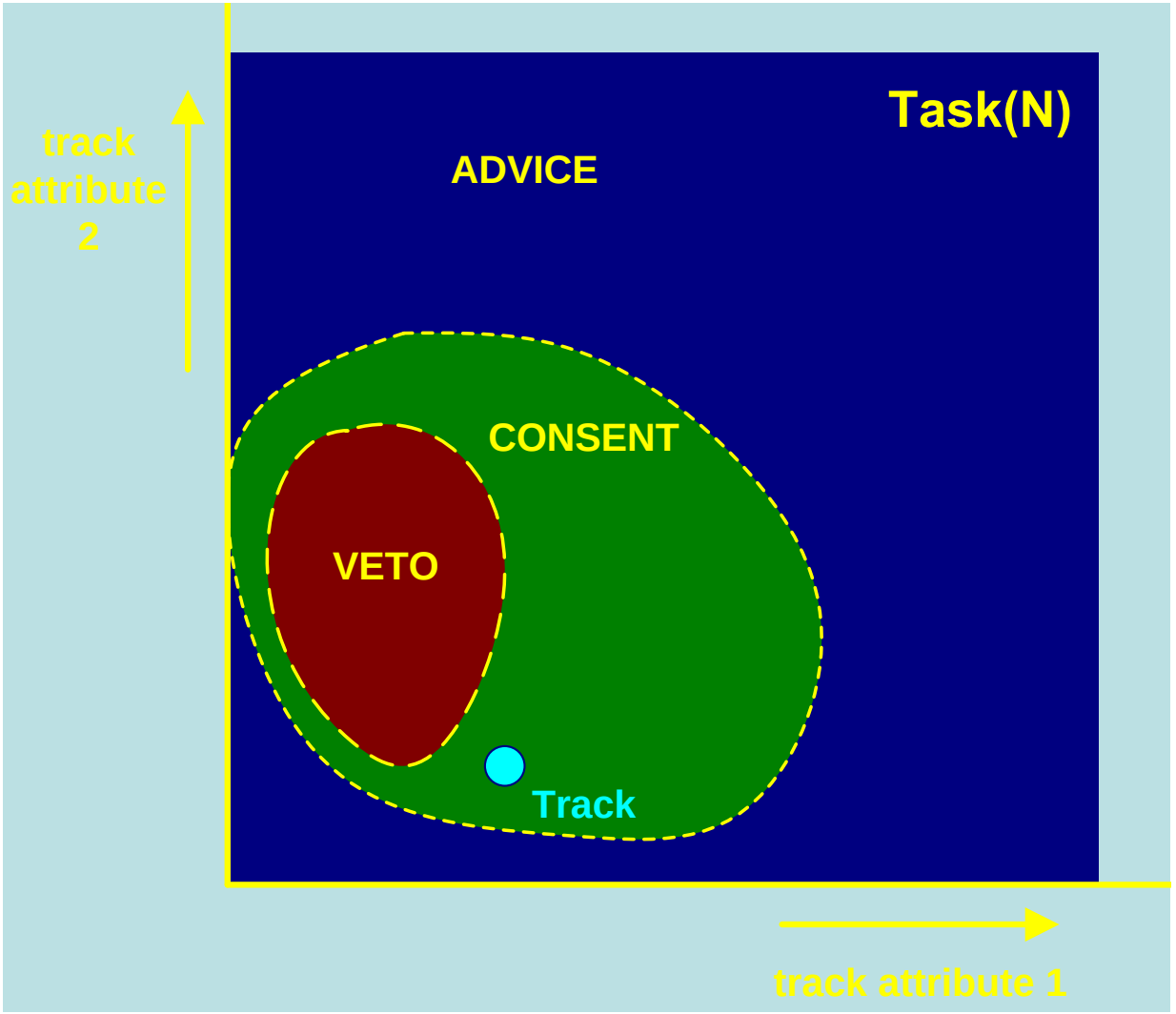
AA Case: System & User World Views in the CMS



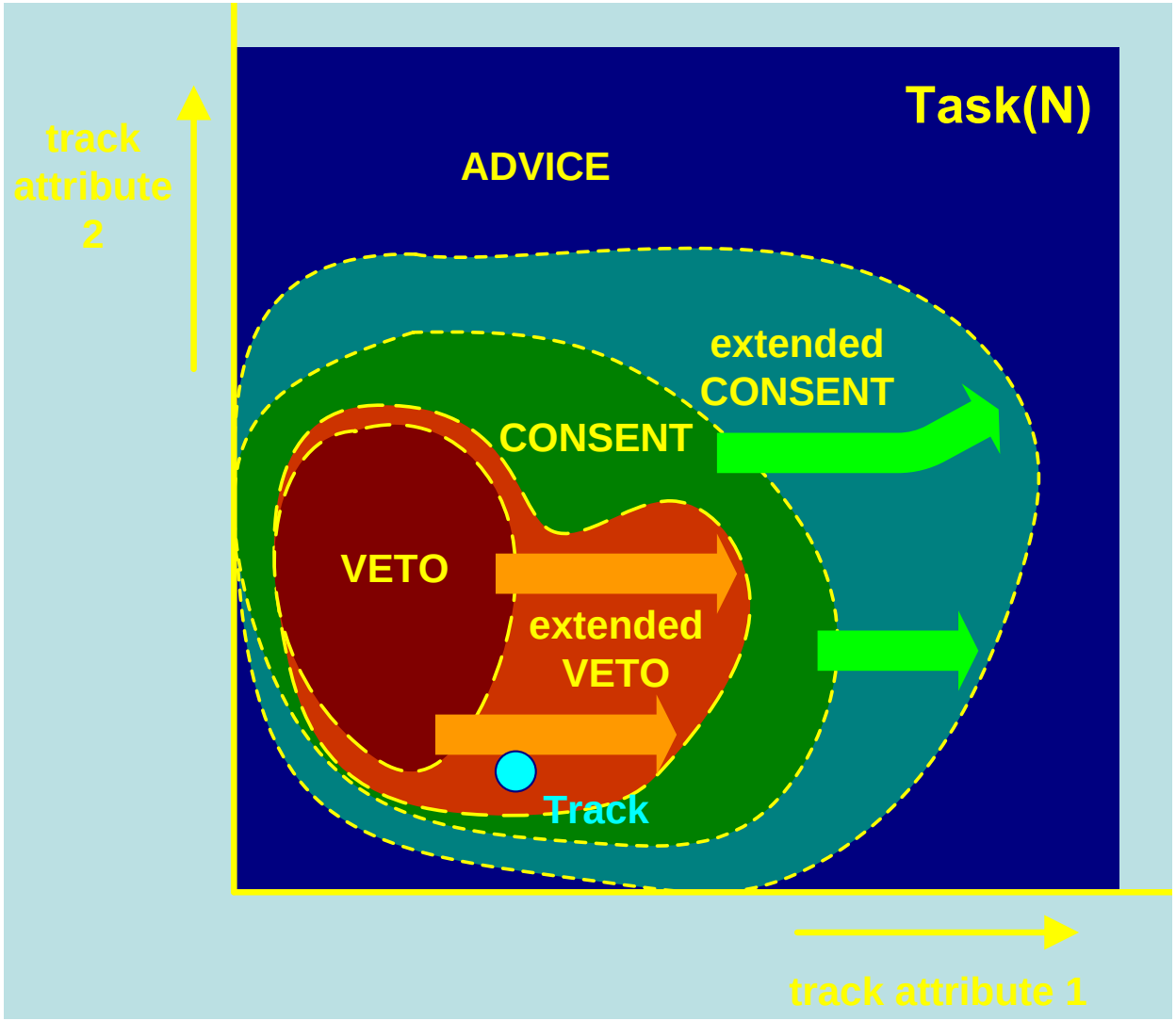
Automation Levels

Automation Level	Division of Labour	System writes to User Space	Signal User
MANUAL	***No system support***	No	No
ADVICE	***"Pull" information***	Manual	No
CONSENT	***"Push" information***	Manual	Yes
VETO	***Supervised system action***	Automatic	Yes
SYSTEM	***Full delegation***	Automatic	No

Automation Levels definable *per Track per Task*



Adaptability is making the Track Sets adjustable



Specify: Core Functions

- Prevent out-of-the-loop problems
- Maintain situational awareness
- Prevent skill degradation
- Ensure Transparency
- Prevent mode errors or automation surprises
- Prevent undesirable system behavior
- Prevent misplaced salience
- Prevent complexity creep
- Prevent/limit increases of system demands
- Prevent errant mental model of the situation
- Prevent unwanted modes of operation
- Prevent tunnel vision
- Extend human memory capacity
- Accommodate workload, stress & fatigue
- Prevent data overload
- Prevent cognitive lockup
- Prevent change blindness
- Improve performance

Specify: Claims

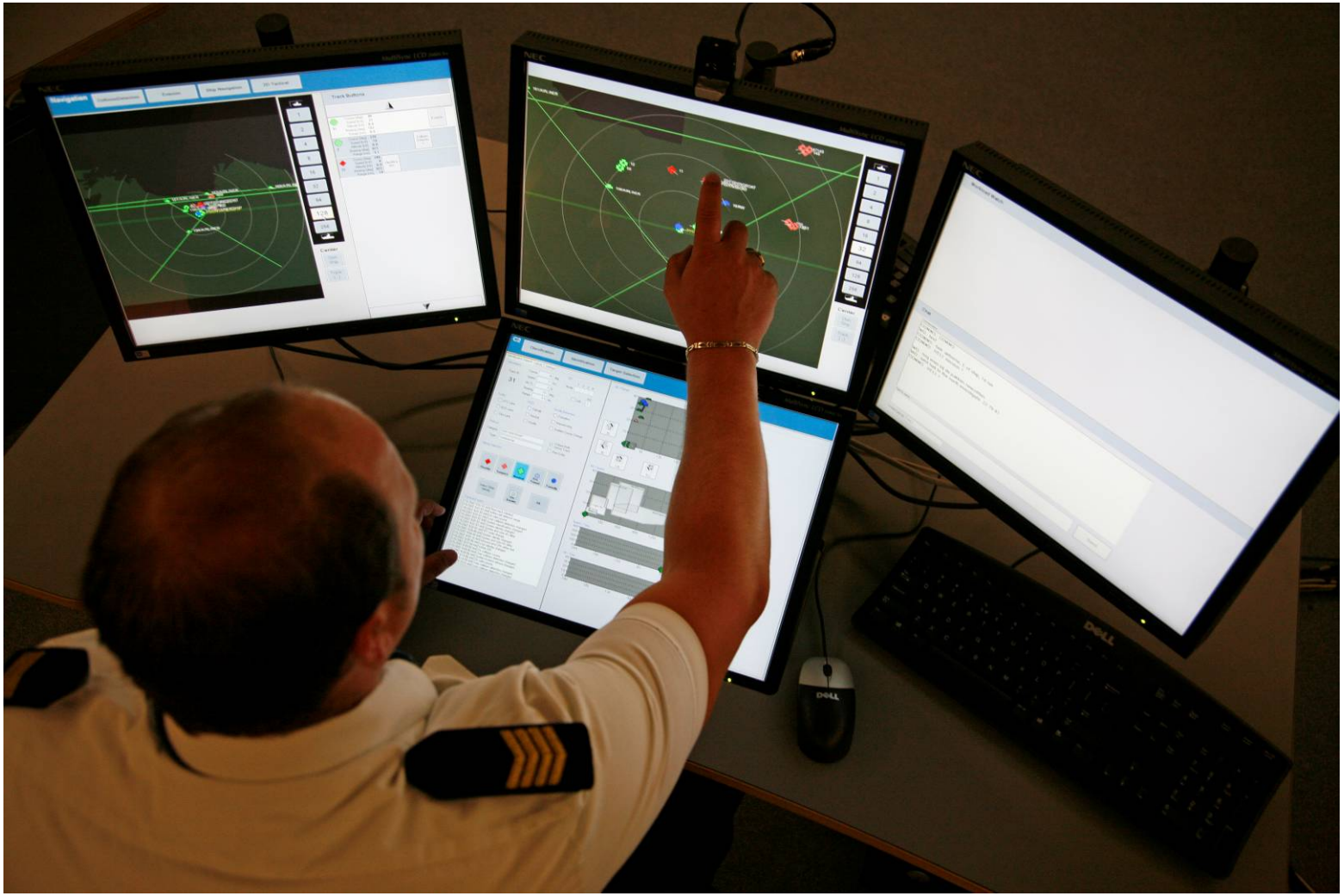
- Core function: Prevent out-of-the-loop problems

- Claim 1:
  - *Feature*: When the task load decreases (e.g. few tracks to handle), a lower level of automation is triggered.
  - *Result*: The user does (almost) everything and handles more tracks, so that (s)he is sufficiently engaged in the current operation (e.g., adequate eye movements and medium arousal level), detects relevant objects in time (e.g., adequate identification performance) and is not involved in unrelated and irrelevant activities (e.g., mainly task-related behavior).

Specify: Scenarios/ Use cases

[UC_Nr] [UC_name]	*Example:* UseCase 3 Increasing SA after decreasing LOA
Goal	Limit out-of-the-loop problems
Actor	Team member of Command and Control Centre
Precondition	AA is at the medium or high level; User has a limited view of tracks as some are handled by the system, limiting his situational awareness to 'dangerous' tracks.
Post condition	AA is set at a lower level More tracks will be handled by the user from now on, increasing his or her overall situational awareness.
Trigger	Amount of work (pending tracks, tracks requiring user attention) is below a preset threshold level.
Main Success Scenario	After decrease of automation level, more tracks of multiple categories will be handled by the user In doing so, the user quickly gets good situational awareness.
Alternative Scenario	……
Satisfies claim	Claim 1, Claim 25
Satisfies requirement	Requirement 13

Prototype Development: Basic-T



Prototype evaluation

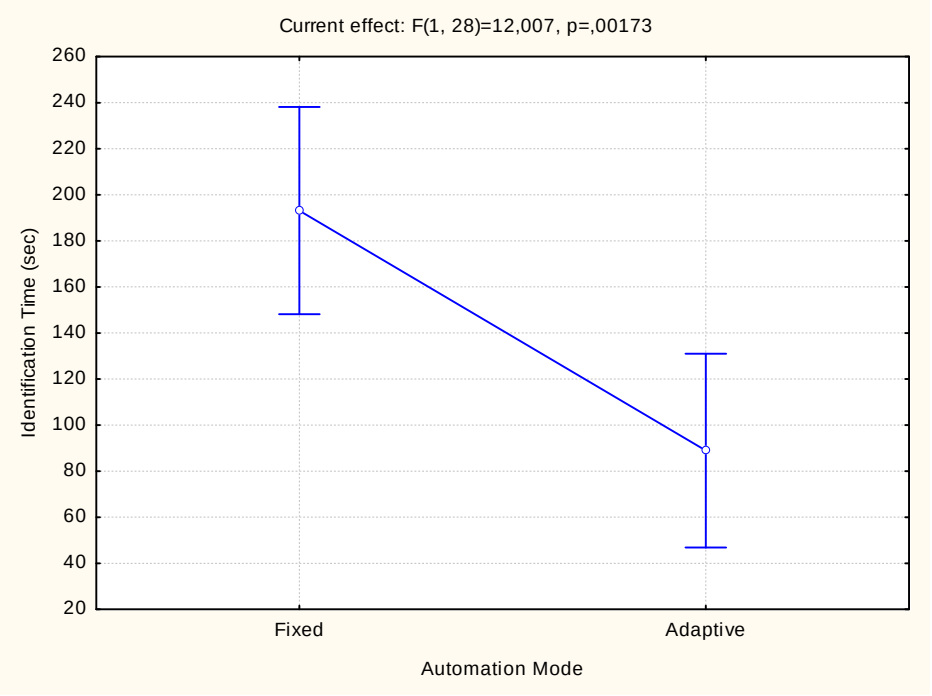
- **Participants**
  - Eight (LVO, CCO, ALVO, ACCO)
  - Two Observers (LVO, CCO)

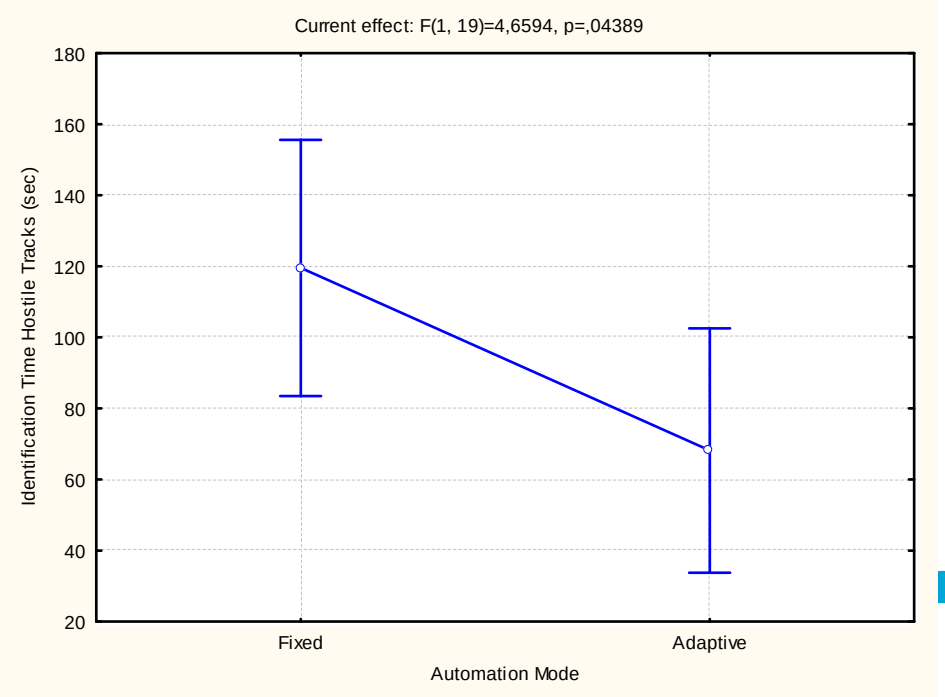
- **Four Scenarios**
  - Two Peace Enforcement Situations close to War (almost ‘classic’)
  - Two Counter Smuggling Situations with Terrorist Threat (‘present-day’)
  - Drafted together with RNlN

- **Automation**
  - Either Fixed (CONSENT mode) or Adaptive (three configurations)
  - Presented to Participants in Varying Configurations

Results Summary

- **Measures of Performance**
  better
- **Observed Situational Awareness**
  unchanged
- **Observed Workload, Correctness**
  unchanged
- **Subjective Workload**
  unchanged





Results and refinements

- Implementation bugs
- Users were sometimes overruled by a system decision after a user decision was already made.

- Core function: Prevent out-of-the-loop problems
- Claim 3.
  - *Feature*: When the automation level is lowered, the user is made aware of tracks that have been handled by the system.
  - *Result*: Tracks that were handled by the system at high automation levels are labeled as such, so that the operator can inspect them after the level of automation has lowered (e.g., user behavior) to improve his or her momentary knowledge of the situation (e.g., adequate situation reports).

So, good IUXE?

- Include Relevant Theory in Root Concept
- Specify the Task Structure
- Define the Context
- Specify the Design Rationale
  - Scenarios & use cases
  - Claims
  - Core functions
- Evaluate to refine and validate the Design Rationale

Literature

Current Lecture (module 2):

- Rosson, M.B. and Carroll, J.M. (2002). Scenario-Based Design. In: J. Jacko & A. Sears (Eds.), The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications. Chapter 53, pp. 1032-1050. Lawrence Erlbaum Associates.

Next Lecture (module 3):

- Paper presentation by student groups

Module 4:

- Neerincx, M.A. (2003). Cognitive task load design: model, methods and examples. In: E. Hollnagel (ed.), Handbook of Cognitive Task Design. Chapter 13 (pp. 283-305). Mahwah, NJ: Lawrence Erlbaum Associates.